Symposium OPEN IVR

VT100 Host Connectivity Configuration and Development Guide

Product release: 4.0
Document release: Standard 1.0
Date: July 1998

© 1998 Northern Telecom
All rights reserved

Printed in the United States of America

Information is subject to change without notice. Northern Telecom reserves the right to make changes in design or components as progress in engineering and manufacturing may warrant.

Aculab is a registered trademark of Aculab Plc. APERTUS EXPRESS is a registered trademark of Computer Network Technology, Inc. DEC and VT420 are registered trademarks of Digital Equipment Corporation. Dialogic is a registered trademark of Dialogic Corporation. Digi is a registered trademark of DIGI International. UNIX is a registered trademark of Novell, Inc. SCO is a registered trademark of The Santa Cruz Operation, Inc. X Window System and X are trademarks of the Massachusetts Institute of Technology.

Publication history

July 1998 This is the Standard 1.0 issue of the *Symposium OPEN IVR VT100 Host Connectivity Configuration and Development Guide*.

Contents

About this guide **vii**

Who should use this guide vii
Contents of this guide. viii
Symposium OPEN IVR 4.0 documentation ix
Conventions used in this guide xii

1 VT100 Gateway overview **1-1**

The VT100 terminal 1-1
VT100 gateway software 1-5

2 VT100 installation **2-1**

Overview 2-1
PC/Xem board 2-2
Installing the VT100 PC/Xem board 2-5
PC/Xe board. 2-7
Installing the VT100 PC/Xe board 2-9
Connecting peripherals 2-11
Connecting to a DCE device (such as a modem) with hardware handshaking 2-11
Connecting the PC/Xe or PC/Xem to a DTE device 2-11
RJ-45 equipped boards 2-13
RS-232 cables and connector options 2-13
Installing and running the MenuPort Interface (MPI) package 2-15
Installing the MPI package. 2-15
Running MPI to configure the PC/Xe or PC/Xem board 2-17
Testing the new ports. 2-23
VT100 PC/Xe board diagnostics 2-25
Removing the VT100 software. 2-26

3 Getting started 3-1

Before using the VT100 gateway . . . 3-1
screen.conf file . . . 3-2
Setting up the trs.conf file . . . 3-5
Example trs.conf file . . . 3-7
Setting up the trs.sdf file . . . 3-9
Setting up the vt100.cfg file . . . 3-11
Defining users and session groups . . . 3-11
Completing your configuration . . . 3-14
Setting up the vt100.ctl file . . . 3-17
Setting up the com.conf file . . . 3-19
A complete sample transaction . . . 3-21
Initial-action template . . . 3-22
Action template performing a transaction . . . 3-25

4 Template files 4-1

Determining the required transactions . . . 4-1
Action templates . . . 4-4
Action template syntax . . . 4-5
Screen templates . . . 4-13
Screen template syntax . . . 4-16
Initial-action templates . . . 4-26

5 Symposium OPEN IVR call flow interface 5-1

Using the COMI, COMO, and COMA cells to access the host computer . . . 5-2
Developing an application that processes terminal sessions . . . 5-3
Setting the COMI cell parameters . . . 5-4
Setting the COMO cell parameters . . . 5-7
Sample application using the COMI, COMO, and COMA cells . . . 5-10

Appendix A: Installing an expander box A-1

Glossary Glossary-1

Index Index-1

About this guide

Who should use this guide

The *Symposium OPEN IVR VT100 Host Connectivity and Development Guide* is intended for use by Symposium OPEN IVR application developers whose voice applications require VT100-based access to computer resources external to the application processor.

This manual assumes that the user is familiar with the operating characteristics of the VT100 terminal, and is experienced in creating voice applications with Symposium OPEN IVR. You should also be familiar with the UNIX operating system and the `vi` text editor (or another text editor installed on your application processor).

Contents of this guide

This guide contains the following chapters:

Chapter 1 VT100 Gateway overview

This chapter provides an overview of the VT100 terminal and a description of the Symposium OPEN IVR VT100 Gateway features.

Chapter 2 VT100 installation

This chapter describes the installation process for both VT100 hardware and software.

Chapter 3 Getting started

This chapter describes the steps you must perform and the additional files you must create to activate the VT100 Gateway software (after installing the VT100 communication hardware and software as described in Chapter 2, “VT100 installation"). This chapter also includes a sample set of template files.

Chapter 4 Template files

This chapter explains how to create the necessary files for sending data to and receiving data from a remote computer.

Chapter 5 Symposium OPEN IVR call flow interface

This chapter explains how to integrate the templates you created in Chapter 4 with your IVR application call flow.

Appendix A - Installing an expander box

This appendix describes how to configure your SCO PC for multiple VT100 terminal sessions.

Symposium OPEN IVR 4.0 documentation

These are the documents in the Symposium OPEN IVR Release 4.0 documentation suite.

Document	NT number/ A0 number P0 number
Symposium OPEN IVR 4.0 Getting Started Guide	NTVE8201 A0732100 P0881271
Symposium OPEN IVR 4.0 Product Guide	NTVE8202 A0732101 P0881273
Symposium OPEN IVR 4.0 System Administration Guide	NTVE8203 A0732102 P0881274
Symposium OPEN IVR 4.0 Application Development Guide	NTVE8209 A0732108 P0881275
Symposium OPEN IVR 4.0 Cell Catalog	NTVE8208 A0732107 P0881276
Symposium OPEN IVR 4.0 Installation, Upgrade, and Maintenance Guide	NTVE8204 A0732103 P0881277
Symposium OPEN IVR Planning and Engineering Guide	NTVE8205 A0732104 P0881278
Symposium OPEN IVR 4.0 Error Messages Guide	NTVE8206 A0732105 P0881279

Document	NT number/ A0 number P0 number
Symposium OPEN IVR 4.0 Web Option Guide	NTVE8210 A0732109 P0881281
Symposium OPEN IVR 4.0 SQL Server User Guide	NTVE8216 A0732115 P0881282
Symposium OPEN IVR 4.0 Service Console Interface Reference Manual	NTVE8207 A0732106 P0881284
Symposium OPEN IVR 4.0 VT100 Host Connectivity Configuration and Development Guide	NTVE8221 A0732120 P0881286
Symposium OPEN IVR 4.0 Text-to-Speech Option Guide	NTVE8213 A0732112 P0881292
Symposium OPEN IVR 4.0 Speech Recognition Option Guide	NTVE8212 A0732111 P0881293
Symposium OPEN IVR 4.0 Fax Option Guide	NTVE8211 A0732110 P0881294
Symposium OPEN IVR 4.0 CAS and ISDN Signaling Option Guide	NTVE8214 A0732113 P0881296

Document	NT number/ A0 number P0 number
Symposium OPEN IVR 4.0 Integration Package for Meridian Link User Guide	P0881297 NTVE8215 A0732114
Symposium OPEN IVR 4.0 Prompt Import Utility Guide	NTVE8218 A0732117 P0881400
Symposium OPEN IVR 4.0 SNA Host Communications Guide	NTVE8227 A0732774 P0881750

Conventions used in this guide

This guide uses several typographic conventions to highlight certain types of information:

- Items that appear on the Symposium OPEN IVR screens are identified in quotes. For example:

  the “Timeout” part of the parameters window

- Prompts, file and menu names, directories, accounts, and options appear in a different typeface. For example:

  `/sci>`

  `sci/install` menu

  `/u/nortel/3270/getbalance.act` file

- Symposium OPEN IVR buffer names appear in all uppercase characters. For example:

  the CURRENT MESSAGE buffer

- Field names appear in italics. For example:

  the *Device* field

- Items you must type appear in bold in a different typeface. For example:

  type `tn5250` at the prompt

- Variables shown in command lines appear in italics. For example:

  the `host_cfg`*n* file, where *n* is a variable representing a board number

- Key names you press appear in angled brackets. For example:

  the <F1> key

- Keyboards usually have keys named <Return> or <Enter> that perform the same function. For convenience, this guide uses the keyname <Enter> to represent both keynames.

Chapter 1: VT100 Gateway overview

This chapter provides an introduction to the Symposium OPEN IVR VT100 Gateway, as well as the following information

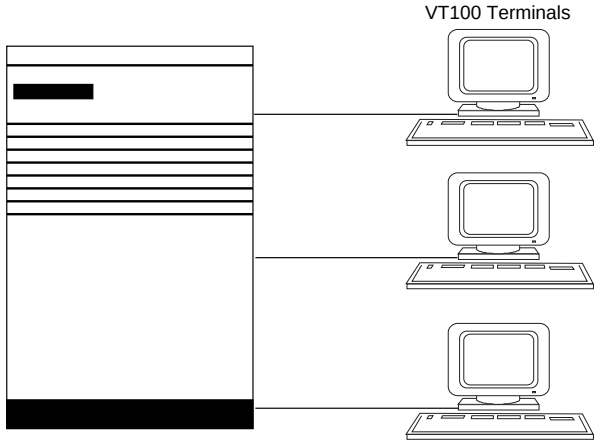
- background information about the VT100 terminal
- descriptions of the Symposium OPEN IVR VT100 Gateway software
- a description of the TRS configuration

Note: Only one host connectivity option is supported. Your system cannot have both SNA host communication and VT100 connectivity.

The VT100 terminal

The VT100 terminal, developed by the Digital Equipment Corporation (DEC), has become one of the most widely used computer terminals in the world. This widespread acceptance makes the VT100 style of host communication a standard for all computer manufacturers (see Figure 1-1).

Figure 1-1: Terminals connected to a host computer

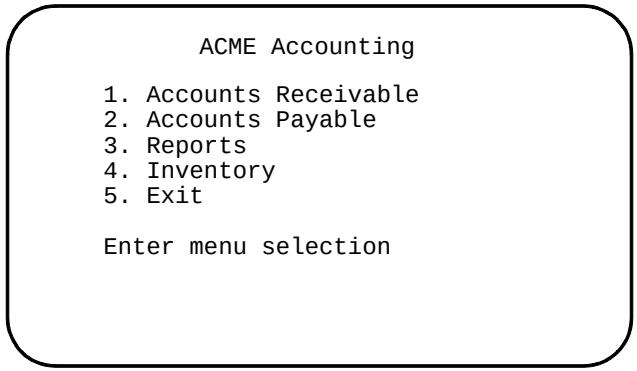


The VT100 terminal uses an asynchronous communication protocol to transmit characters to and from a host computer. The VT100 Gateway communicates with a host through a serial port and a modem, or through a serial port and a direct connection. With respect to the VT100 Gateway product, a host computer is any computer that can accept a VT100 terminal connection, including mainframes, minicomputers, and workstations.

The standard VT100 terminal has a 24x80 character display, and can be accessed nonsequentially. This means the terminal can access text anywhere on the screen. This enables you to delete and insert text, scroll the page, and move the cursor to any position on the screen. Your ability to access the VT100 screen nonsequentially depends on the host application you are using.

Figure 1-2 shows a sample VT100 screen with a host application running.

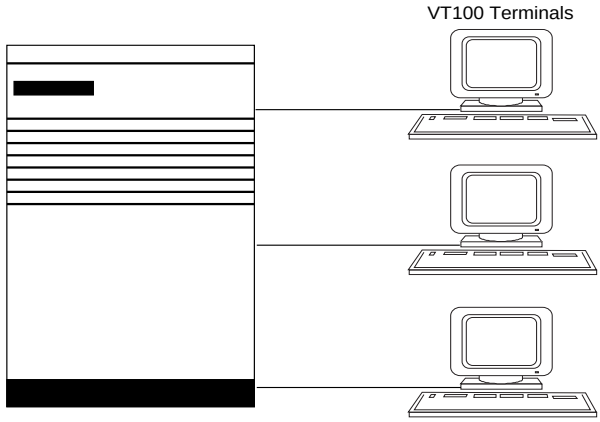
Figure 1-2: Sample VT100 screen



An active host to terminal connection is called a session. The VT100 Gateway can execute a series of transactions during a session. A transaction is the series of steps required to perform a specific function such as finding a customer's account balance. When one transaction finishes, the session is available to execute another transaction. The Symposium OPEN IVR application processor (AP), when configured with the VT100 Gateway product, can control multiple simultaneous sessions with the host computer.

The Symposium OPEN IVR VT100 Gateway product enables an IVR application to establish sessions with a host, just as multiple VT100 terminals would. Using Symposium OPEN IVR and the VT100 Gateway, you can execute any programs or commands that you can execute from a VT100 terminal. Symposium OPEN IVR can store host output in the buffers of an application, then play the information to a caller, as shown in Figure 1-3.

Figure 1-3: Logical terminal sessions running on a host computer



The AP physically connects to the host via a modem if the host is not local, or via a null modem connector if the host resides locally. See the information specific to your platform about connecting with a modem.

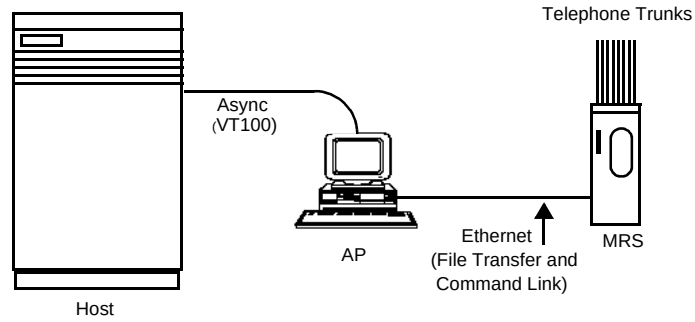
VT100 gateway software

You can install the Symposium OPEN IVR VT100 Gateway on any IVR applications processor (AP). To support the VT100 Gateway, the AP must

- be connected to one or more host computers via an asynchronous connection
- have enough serial ports to provide enough terminal connections (see the documentation specific to your platform for information about terminal connections)

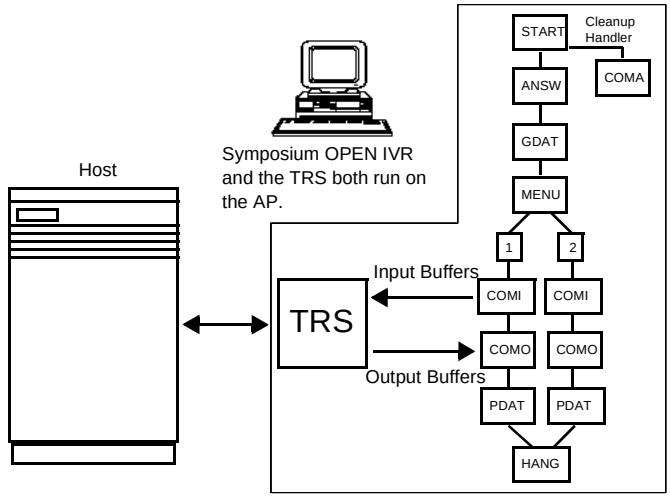
Nortel does not supply the VT100 communication hardware. Figure 1-4 shows the typical hardware configuration for processing VT100 terminal sessions.

Figure 1-4: Symposium OPEN IVR VT100 Gateway configuration



A Symposium OPEN IVR process called the Terminal Resource Server (TRS) controls all VT100 sessions, and also manages all host connections. The TRS runs as a stand-alone process within the Symposium OPEN IVR architecture, and starts when Symposium OPEN IVR is started. To use the TRS, place a COMI cell in the Symposium OPEN IVR call flow at the point where you need to establish a host connection. The COMI cell sends requests for information to the TRS process, which then passes them on to the host. A COMO cell retrieves the information sent to the TRS process by the host, and then ends the transaction. A COMA cell aborts the host transaction in the case of a hang-up or an error. This process is shown in Figure 1-5.

Figure 1-5: The COM cells communicate with the Terminal Resource Server



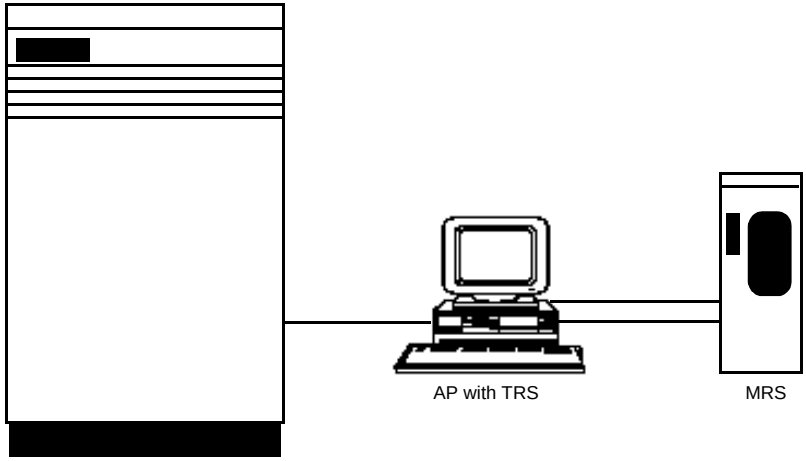
The TRS process uses template files (described in Chapter 2, “VT100 installation", and Chapter 4, “Template files") to control the communication process between the IVR application and the host computer.

As illustrated in Figure 1-5, the TRS uses the data passed to it by the application as a signal to establish a terminal session. The TRS then controls the screen display associated with the host computer.

TRS configuration

You install all of the hardware and software required to run the VT100 Gateway product on the AP. Figure 1-6 illustrates the TRS configuration.

Figure 1-6: TRS configuration



Chapter 2: VT100 installation

Overview

Two types of boards are available for connectivity to a VT100 host with speeds of up to 115.2 Kbps. The PC/Xem board is used for connectivity when 16 to 32 ports are required. The PC/Xe board is used for connectivity when a maximum of eight ports is required.

The PC/Xem board provides up to 32 RS-232 ports by connecting to an external port expander called a PORTS module, which provides 16 DB-25 or RJ-45 connectors. Additional PORTS modules can be daisy-chained to provide up to 32 RS-232 ports.

> ***Note:*** The PC/Xem board is supported on Symposium OPEN IVR release 4.0 and release 2.x platforms.

The PC/Xe board provides eight RS-232 ports. The board has a DB78 connector which connects to a cable which fans out to eight connectors.

> ***Note:*** The PC/Xe board is supported on Symposium OPEN IVR release 2.x platforms only.

PC/Xem board

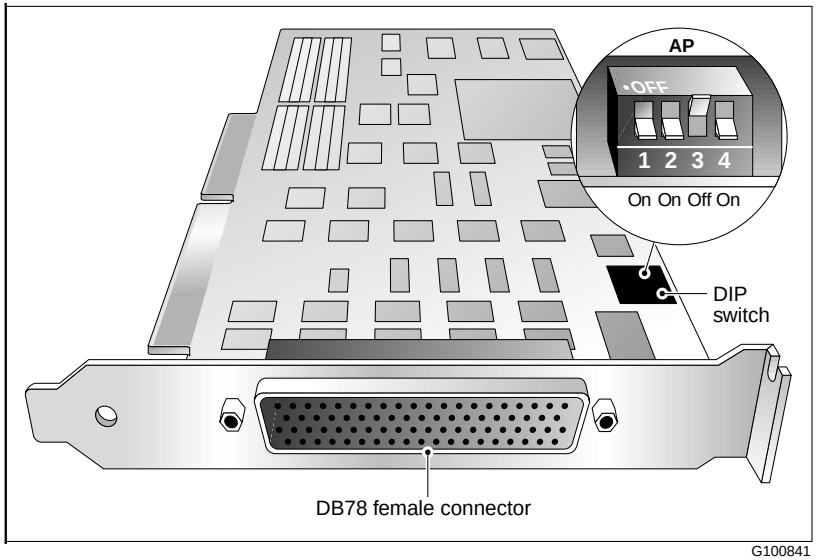
The PC/Xem board is a full-length card with a 20 MHz IDT 3051 RISC processor. The device driver software controls the PC/Xem board through an I/O port mapped to the computer's I/O bus. The board must have its own I/O port address, not used by any other device in the computer.

The switch settings for the different I/O address ranges are shown in Table 2-1.

Table 2-1: PC/Xem board I/O address switch settings

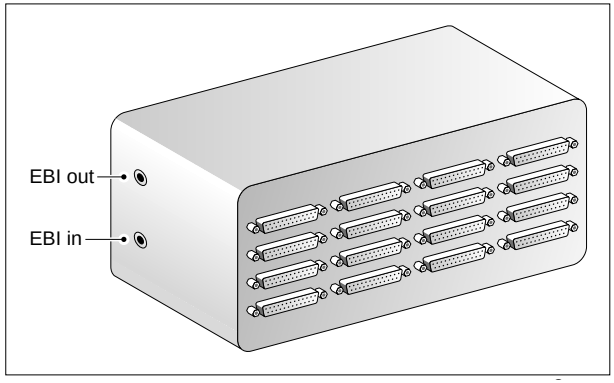
I/O Address	1	2	3	4
104h	OFF	OFF	ON	ON
114h	OFF	ON	OFF	ON
124h	OFF	ON	ON	ON
204h	ON	OFF	OFF	ON
224h	ON	OFF	ON	ON
304h	ON	ON	OFF	ON (<-AP)
324h	ON	ON	ON	ON

Figure 2-1: VT100 PC/Xem board and switch settings



The PC/Xem host adapter provides connections for up to 32 ports utilizing the PORTS module. Two PORTS modules can be linked, giving the capability of serving up to 32 users from a single host slot. The PORTS module uses either RJ-45 or DB-25 connectors.

Figure 2-2: DB-25 PORTS module



Each DB-25 PORTS module is labeled with the appropriate channel number and uses EIA RS-232 voltage levels. Signals are assigned to the pins as specified in Table 2-2.

Table 2-2: Pin signals

PIN#	Signal	Description
2	TXD	Transmit Data
3	RXD	Receive Data
4	RTS	Request to Send
5	CTS	Clear to Send
6	DSR	Data Set Ready
7	SG	Signal Ground
8	DCD	Data Carrier Detect
20	DTR	Data Terminal Ready
22	RI	Ring Indicator

A female HD-44 connector connects the first PORTS module to the PC/Xem board using a 5 ft. (1.5 m) cable. Additional PORTS modules are connected to the first module using a 16 in. (40.6 cm) HD-44 cable.

Installing the VT100 PC/Xem board

To install the PC/Xem board, follow these steps.

CAUTION

Risk of electrostatic discharge

When you install the PC/Xem adapter, avoid damage from electronic discharge by using a grounding strap, by keeping the board in its protective package until you are ready to install it, and by handling the board by the edges only.

1. Shut down the AP or MRS/AP by following the established shutdown procedures.
2. Turn off power to the system, and disconnect the power cord.
3. Remove the side cover.
4. The board is installed in EISA slots 1 or 2, or both.

   Note: The bottom nine slots are EISA slots. Slot 10 is the CPU slot, and the top four slots are PCI slots. Slot 7 is also a CPU slot, but it is used as an EISA slot.

5. Remove the slot cover, and save the screw for later use.

6 On an AP (not an MRS/AP), set the I/O DIP switch to I/O address 114 by setting the switches to the following:

— switch 1 - off

— switch 2 - on

— switch 3 - off

— switch 4 - on

On an MRS/AP (not an AP), set the I/O DIP switch to I/O address 114 by setting the switches to the following:

— switch 1 - off

— switch 2 - on

— switch 3 - off

— switch 4 - on

Note: There are two jumpers on the PC/Xem board. These jumpers should be left in the factory set position, between pins 1 and 2 (the top two pins).

7 Firmly push the adapter into the connector of the available slot. Do not force the adapter into the slot; it should slide in easily and if it does not, it may be misaligned. Secure the adapter with the screw from Step 5.

8 Reinstall the side cover.

9 Connect the cable from the board to the PORTS module.

10 Reconnect the power cord, and power up the system.

The hardware installation is complete. Your software must still be installed.

PC/Xe board

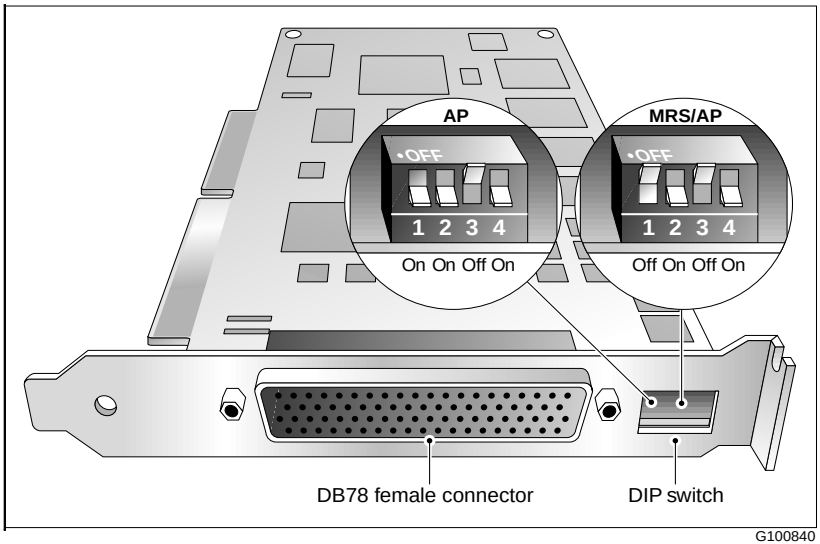
The PC/Xe board has four 8-bit I/O registers which the system uses for board configuration. The four registers occupy consecutive I/O addresses, the first of which is defined by setting switches 1 to 3 on DIP switch DS1. If you are installing multiple PC/Xe boards, each board must have its own I/O address.

The switch settings for the different I/O address ranges are shown in the following table.

Table 2-3: PC/Xe I/O address switch settings

I/O Address	1	2	3	4
100h - 103h	OFF	OFF	ON	ON
110h - 113h	OFF	ON	OFF	ON -> MRS/AP
120h - 123h	OFF	ON	ON	ON
200h - 203h	ON	OFF	OFF	ON
220h - 223h	ON	OFF	ON	ON
300h - 303h	ON	ON	OFF	ON -> AP
320h - 323h	ON	ON	ON	ON

Figure 2-3: VT100 PC/Xe board and switch settings



Installing the VT100 PC/Xe board

To install the PC/Xe board, follow these steps.

CAUTION

Risk of electrostatic discharge

When you install the PC/Xe adapter, avoid damage from electronic discharge by using a grounding strap, by keeping the board in its protective package until you are ready to install it, and by handling the board by the edges only.

1 Shut down the AP or MRS/AP by following the established shutdown procedures.

2 Turn off power to the system, and disconnect the power cord.

3 Remove the side cover.

4 The board is installed in EISA slots 1 or 2 or both.

Note: Note: EISA slot 1 is at the bottom of the motherboard, slot 2 is above slot 1, and so on.

5 Remove the slot cover, and save the screw for later use.

6 On an AP (not an MRS/AP), set the I/O DIP switch to I/O address 100 by setting the switches to the following:

— switch 1 - off

— switch 2 - off

— switch 3 - on

— switch 4 - on

On an MRS/AP (not an AP), set the I/O DIP switch to I/O address 300 by setting the switches to the following:

— switch 1 - on

— switch 2 - on

— switch 3 - off

— switch 4 - on

7 Firmly push the adapter into the connector of the available slot. Do not force the adapter into the slot; it should slide in easily and if it does not, it may be misaligned. Secure the adapter with the screw from Step 5.

8 Reinstall the side cover.

9 Connect the DB-78 cable, which fans out to eight DB-25 connectors, to the PC/Xe board.

10 Reconnect the power cord, and power up the system.

The hardware installation is complete. Your software must still be installed.

Connecting peripherals

Connecting to a DCE device (such as a modem) with hardware handshaking

A 10-pin RJ-45 plug may be difficult to obtain in the retail market; therefore, most PC/Xe or PC/Xem device driver software incorporates an optional feature called ALTPIN, which swaps the logical functions of Data Set Ready (DSR) with Data Carrier Detect (DCD). When ALTPIN is enabled (see your device driver software reference manual for instructions), DCD becomes available on pin 1 of an 8-pin RJ-45 connector (equivalent to pin 2 of a 10-pin connector). If you wish to build an 8-wire modem cable for an RJ-45 equipped board, use an 8-pin RJ-45 plug wired as shown in the following table.

RJ-45 8-Pin	**DB-25 Male**
DCD (1)	(8) DCD
RTS (2)	(4) RTS
GND (3) {CABLE SHIELD}(1) GND	
TxD (4)	(2) TxD
RxD (5)	(3) RxD
SG (6)	(7) SG
CTS (7)	(5) CTS
DTR (8)	(20) DTR

Connecting the PC/Xe or PC/Xem to a DTE device

A DTE device is a terminal, serial printer, another computer's serial port, and so on. To connect the PC/Xe (which are also DTE devices) to another DTE device, you need a null modem cable or adapter.

Software handshaking (XON/XOFF)

In most cases, serial terminals and printers need only a three-wire connection to the PC/Xe board. All PC/Xe or PC/Xem device driver software supports XON/XOFF (software) handshaking, so the only signal lines necessary are Transmitted Data (TxD), Received Data (RxD), and Signal Ground (SG). It may be necessary to disable Data Carrier Detect (DCD) sensing through a software command. For instructions, see your PC/Xe or PC/Xem device driver software manual.

Cables must be shielded to remain in compliance with FCC certification requirements, and the shield should be connected to Chassis Ground (GND) at both ends of the cable run. A simple cable for connecting a terminal or a printer to a DB-25 equipped board is shown in the following table. The female DB-25 end plugs directly into one of the DB-25 connectors on the fanout cable or connector box assembly.

DB-25 Female	DB-25 Male
TxD (2)	(3) RxD
RxD (3)	(2) TxD
RTS (4)	(4) RTS
CTS (5)	(5) CTS
DSR (6)	(6) DSR
GND (7)	(7) GND
DCD (8)	(8) DCD
DTR (20)	(20) DTR
RI (22)	(22) RI
GND (Shell){CABLE SHIELD}(1) GND	

RJ-45 equipped boards

Software handshaking (XON/XOFF)

In most cases, serial terminals and printers need only a three-wire connection to the PC/Xe or PC/Xem board. All PC/Xe or PC/Xem device driver software supports XON/XOFF (software) handshaking, so the only signal lines necessary are Transmitted Data (TxD), Received Data (RxD), and Signal Ground (SG). It may be necessary to disable Data Carrier Detect (DCD) sensing through a software command: see your PC/Xe or PC/Xem device driver software manual for instructions. Cables must be shielded to remain in compliance with FCC certification requirements, and the shield should be connected to Chassis Ground (GND) at both ends of the cable run. A simple cable for connecting a terminal or a printer to an RJ-45 equipped PC/Xe or PC/Xem is shown in the table below.

RJ-11 4-Pin	DB-25 Male
GND (1) {CABLE SHIELD} (1) GND	
TxD (2)	(3) RxD
RxD (3)	(2) TxD
GND (4)	(7) GND
GND (Shell){CABLE SHIELD}(1) GND	

RS-232 cables and connector options

Cabling

RS-232 serial interface cables should be shielded, low capacitance cables, ideally designed specifically for serial data transmission. The shield should be grounded at both ends of the cable. Chassis Ground is available on the shell of the DB-25 and DB-9 connectors, and pin 4 of the 10-pin RJ-45 connector is ideal for this purpose.

Environment

While good shielding provides reasonable protection against electro-magnetic nterference, or EMI, also referred to as noise, cables should still be routed away from noise sources wherever possible. Avoid laying cables near transformers, generators, motors, fluorescent lights, and so on.

Capacitance versus length of run

The total capacitance of a cable affects the integrity of transmitted data. Serial interface cable is usually rated in pico Farads per foot. As a rule of thumb, the total capacitance of a cable (including the connectors) should not exceed 2500 pF. Therefore, if a cable has a capacitance of 50 pF/ft, and the connectors are 100 pF each, the maximum recommended cable length is 46 feet (1.4 m). If the cable is rated at 12.5 pF/ft, the maximum recommended cable length is 184 feet (55.2 m), and 5 pF/ft cable can be run up to 460 feet (138 m). In situations where low-capacitance cable is unavailable, or very long cable runs are required, short-haul modems can be used to increase the effective range of the RS-232 interface. Short-haul modems are similar to standard modems, except that they are connected directly to each other using a cable instead of going through a telephone circuit.

> ***Note:*** Use only externally powered short-haul modems with the PC/Xe or PC/Xem board.

Installing and running the MenuPort Interface (MPI) package

The PC/Xe or PC/Xem device driver software for SCO-UNIX is installed by the MenuPort Interface package. Installation of the PC/Xe or PC/Xem device drivers is a two-step procedure. First, the MPI package is installed onto the hard disk. Then, the MPI is run, and the appropriate device drivers for the AP or MRS/AP are installed and configured.

Installing the MPI package

To install the MPI package, follow these steps.

1. Log in to the AP as **`nortel`**.
2. Click on the UNIX window.
3. Type **`su`** and press <Enter>.
4. Enter the root password at the prompt.
5. Insert the DigiWARE diskette labeled SCO UNIX into the AP.
6. Type **`custom`** to run the custom program.
7. With `Install` highlighted, press <Enter>.
8. With the cursor bar in the *Select a Product* field and an asterisk (*) next to `* A New Product`, press <Enter>.
9. The cursor proceeds to the *Choose an Option* field. With the entire product highlighted, press <Enter>.

   The system displays the following message:

```
Insert the requested volume and press <Return> to
continue the installation

Insert:    Distribution

             Floppy Volume I

Continue  Quit
```

10 Press <Enter>.

The system displays the following message:

```
Insert the custom data files ...
Insert the requested volume and press <Return> to
continue the installation

Insert:    MenuPort Interface (MPI) Package

            Floppy Volume 1

Continue  Quit
```

11 Press <Enter>.

The following prompt appears:

```
Insert: MenuPort Interface (MPI) Package Floppy
Volume 1.
```

12 Press <Enter>.

The following messages appear:

```
Extracting files ...

Executing MenuPort Interface (MPI) Package Init Script
The MenuPort Interface (MPI) software has been
installed into your system. MPI must be invoked for the
Xe, Xi, Xr  EPC/X, C/X or Xem installation.
To use MPI, please type ‘mpi’ from the command line.

Press any key to continue
```

13 Press <Enter>.

The `Custom Products Currently Installed` screen reappears.

14 Type **q** to quit, and then press <Enter> to exit.

15 Remove the floppy disk from the AP.

The MPI package installation is complete. Now you must run MPI to configure the board.

Running MPI to configure the PC/Xe or PC/Xem board

To configure the board using MPI, follow these steps.

1 To run the MPI program, from the # prompt, type **mpi**

Note: If the cursor keys do not appear to be functioning correctly, log off by typing **exit** and log in again. When the message TERM = (ansi) appears, type **vt320** and then begin at step 1 again.

2 After the restricted legend appears, press <Enter> .

The system displays the following main menu.

```
                    Main
Config   - Install, Change, and Remove
System   - Build, Update, and Image
Monitor  - Digi Port Authority (DPA)
Ports     - Port Configuration Utility (PCU)
Notes     - Release Notes

F1=Help  ArrowKeys=Select <ENTER>=Accept <ESC>=Quit
```

3 With the cursor on Config, press <Enter>.

The system displays the following.

```
Install   - Install New Products
Change     - Change/Show Installed Products
Remove      - Remove Installed Products

F1=Help  ArrowKeys=Select <ENTER>=Accept <ESC>=Quit
```

4 With the cursor on `Install`, press <Enter>.

The system displays the following.

```
                    Pick List
Digi Xe and Xi Family
Digi Xr Family
Digi EPC/X, C/X, and Xem Family


F1=Help  ArrowKeys=Select <ENTER>=Accept <ESC>=Quit
```

5 If installing the PC/Xe board, place the cursor on `Digi Xe and Xi Family`, and press <Enter>.

If installing the PC/Xem board, place the cursor on `Digi EPC/X, C/X and Xem Family`, and press <Enter>.

The system displays the following.

```
Board Configuration
Total number of boards to be installed: 1
Board Type: Digi_8e (8K,RS232/RS422)
I/O Address: 0x100
 Memory Address: 0x0D0000
Asynchronous TTY Ports:  /dev/ttyi1a - /dev/ttyi1h
With Modem Control:  /dev/ttyi1A - /dev/ttyi1H
DigiPrint Ports:  /dev/pri1a - /dev/pri1h
With Modem Control:  /dev/pri1A - /dev/pri1H
Space= NextChoice  ArrowKey=NextField  <Esc>=Abort
F1=Help  F2=Choices F4=ShowBoard  F5=Continue
```

6 Press <Enter> until the cursor is on the *I/O Address* field.

7 Press <F2> to display the list of choices.

Note: If the <F2> key does not appear to be functioning correctly, log off by typing **`exit`** and log in again. When the message `TERM = (ansi)` appears, type **`vt320`** and then begin at step 1 again.

8 If installing the PC/Xe board

- on an AP, place the cursor on 0x100 and press <Enter>
- on an MRS/AP, place the cursor on 0x100 and press <Enter>

If installing the PC/Xem board

- on an AP, place the cursor on 0x114 and press <Enter>
- on an MRS/AP, place the cursor on 0x114 and press <Enter>

9 Move the cursor to `Memory Address` and press <F2> to list the choices.

10 Whether this is the PC/Xe or PC/Xem board, on an AP, place the cursor on `0x0D0000` and press <Enter>.

Whether this is the PC/Xe or PC/Xem board, on an MRS/AP, place the cursor on `0x0DC000` and press <Enter>.

11 Press <F5>. If <F5> does not work, to simulate pressing <F5>, hold down the <Control> key and press the <F> key. Remove your hand from the keyboard and press the <5> key along the top row of the keyboard.

The system displays the following.

```
Dscreen/Altpin
Do you wish to install DigiScreen? Yes
Number of DigiScreen devices:  32
Altpin setting: Off

Space=NextChoice  ArrowKey=NextField  <ESC>=Abort

F1=Help          F5=ConfigKernel
```

12 Press <F5>. If <F5> does not work, to simulate pressing <F5>, hold down the <Control> key and press the <F> key. Remove your hand from the keyboard and press the <5> key along the top row of the keyboard.

The system displays the following.

```
Configuring dscr device driver ...
Creating Space.c file ...
Creating Master file ...
Creating System file ...
Major device number is 79.
Creating Node file ...
Installing dscr driver ...
Dscr driver configuration completed.

Press any key to continue:
```

13 Press <Enter> to continue.

The system displays the following.

```
Configuring a stub driver for dps ...
Creating Master file ...
Creating System file ...
Major device number is 80 ...
Creating Node file ...
Installing pcx1 driver ...
DPA stub driver configuration completed.

Please press any key to continue:
```

14 Press <Enter> to continue.

The system displays the following.

```
Configuring Digi X* device driver ...
Creating Space.c file ...
Creating Master file ...
Creating System file ...
Major device number is 81
Creating Node file ...
Removing IXANY from /etc/gettydefs ...
Removing Digi X* entries from /etc/ttytype ...
Adding Digi X* entries to /etc/inittab and
/etc/ttytype ...
Installing Digi X* driver ...

Digi X* device driver configuration completed.

Do you want to rebuild the kernel no (y/n) ?
```

15 Type **y** and press <Enter>.

The system displays the following.

```
Do you really want to rebuild the kernel now (y/n)?
```

16 Type **y** and press <Enter>.

The system displays the following.

```
Re-linking the kernel ...

The UNIX Operating System will now be rebuilt.
This will take a few minutes. Please wait ...

Backing up /unix to /unix.old
Installing new unix
Setting up the new kernel environment
The kernel has been successfully linked and installed.
To activate it, reboot your system.

Press any key to continue:
```

17 Press <Enter>.

The `Config` menu reappears.

18 Press <ESCAPE> twice to quit the MPI program.

19 The tty ports must be activated using the following commands:

PC/Xem board		PC/Xe board
(1st 16 ports)	**(2nd 16 ports)**	enable ttyi1a
enable ttya01	enable ttyb01	enable ttyi1b
enable ttya02	enable ttyb02	enable ttyi1c
enable ttya03	enable ttyb03	enable ttyi1d
enable ttya04	enable ttyb04	enable ttyi1e
enable ttya05	enable ttyb05	enable ttyi1f
enable ttya06	enable ttyb06	enable ttyi1g
enable ttya07	enable ttyb07	enable ttyi1h
enable ttya08	enable ttyb08	
enable ttya09	enable ttyb09	
enable ttya10	enable ttyb10	
enable ttya11	enable ttyb11	
enable ttya12	enable ttyb12	
enable ttya13	enable ttyb13	
enable ttya14	enable ttyb14	
enable ttya15	enable ttyb15	
enable ttya16	enable ttyb16	

20 Reboot the AP. Type **`init 6`**.

The MPI configuration is complete.

Testing the new ports

To test VT100 ports for host connectivity, follow these steps.

1 Connect a terminal to the ports (using a null modem, if necessary). By default, the configuration parameters in the VT100 software are set to Full Duplex, 8 data bits, No Parity, and 1 Stop Bit. Set the terminal or emulation software appropriately.

2 To test the connections to each terminal, type the following command at the AP console for each port added:

```
date > /dev/ttyi1a
```

Note: This assumes that the terminal is connected to `ttyi1a`.

3 What happens when you connect?

- If the date appears on the terminal's screen, the device is properly connected.
- If the date does not appear on the terminal's screen, then that terminal is not receiving data; check the power, cables, connection, and so on.
- If everything appears to be connected properly but you are still not getting output, run the loopback tests described in your SCO-UNIX documentation.
- If nonsense characters are printed on the terminal's screen, check the baud rate, data bits, stop bits, and parity setting on your terminal.

4 To change the settings on the AP, edit the `/etc/inittab` file for the specific `tty` port.

```
Sample /etc/inittab line


c03:2:respawn:/etc/getty ttyi1a 9600
```

Note: You change the settings for a specific `tty` port by changing the field that follows the `ttyi1a` field. This field (9600 in the sample above) refers to a line in the `/etc/gettydefs` file which begins with this value. The 9600 line in `/etc/gettydefs` contains many parameters, including the data bits, parity, and so on.

By changing the `/etc/inittab` file to refer to a different line in the `/etc/gettydefs` file, the characteristics of the port are changed. For example, changing the 9600 in the `/etc/inittab` file to the word console would refer to the characteristics defined by the line in the `/etc/gettydefs` file.

5 After making a change to the `/etc/inittab` file, disable and then enable the changed port. To do so, type the following:

```
disable ttyi1a

enable ttyi1a
```

CAUTION

Risk of data loss

Do not use MPI to change the port settings. Earlier VT100 software releases removed the `/etc/inittab` file when MPI was used to adjust the port settings**.**

VT100 PC/Xe board diagnostics

The board's manufacturer provides a diagnostics disk. To run the user diagnostics, follow these steps.

1. Boot the AP using an MS-DOS 6.0 or higher version system disk.
2. Type **`a:\diags\ud-cisc`** to start the diagnostics.
3. When asked for a board family, press <a>.
4. Select the ports to be tested.
5. Enter at least two parameters:
   - the board's I/O address, set on the DIP switches:
     - 0x300 - 0x303 for the 700t AP
     - 0x300 - 0x303 for the 700t MRS/AP
   - the host base address (the starting address of the board's dual-ported memory)
     - 0xD8000 - 0xD9FFF for the 700t AP
     - 0xD8000 - 0xD9FFF (or 0xE0000 - 0xE1FFF if a VR/160 is installed) for the 700t MRS/AP

   or execute the memory map utility `digimmap.exe` to find a good address to use
6. Press <e> to start the tests.

   Note: The tests run consecutively, and pass/fail status is indicated on the right-hand side of the screen.

   If all tests pass, the board is functioning correctly, and you are ready to install the device driver software.

   If failures occur, the most likely cause is a memory conflict. Try a different host base address, and execute the diagnostic again. If you get ahardware reset error, try a different I/O address. (Be sure to set the DIP switches for the new address.)

Removing the VT100 software

To remove the VT100 software, perform the following steps.

1 All the tty ports which were activated must be deactivated using the following commands:

PC/Xem board		**PC/Xe board**
(1st 16 ports)	**(2nd 16 ports)**	disable ttyi1a
disable ttya01	disable ttyb01	disable ttyi1b
disable ttya02	disable ttyb02	disable ttyi1c
disable ttya03	disable ttyb03	disable ttyi1d
disable ttya04	disable ttyb04	disable ttyi1e
disable ttya05	disable ttyb05	disable ttyi1f
disable ttya06	disable ttyb06	disable ttyi1g
disable ttya07	disable ttyb07	disable ttyi1h
disable ttya08	disable ttyb08	
disable ttya09	disable ttyb09	
disable ttya10	disable ttyb10	
disable ttya11	disable ttyb11	
disable ttya12	disable ttyb12	
disable ttya13	disable ttyb13	
disable ttya14	disable ttyb14	
disable ttya15	disable ttyb15	
disable ttya16	disable ttyb16	

2 From the # prompt, type: **`mpi`**

Note: If the cursor keys do not appear to be functioning correctly, log off by typing **`exit`** and log in again. When the message `TERM = (ansi)` appears, type **`vt320`** and then begin at step 1 again.

3 After the restricted legend appears, press <Enter> .

The Main Menu displays the following.

```
                                Main
Config   - Install, Change, and Remove
System   - Build, Update, and Image
Monitor   - Digi Port Authority (DPA)
Ports      - Port Configuration Utility (PCU)
Notes     - Release Notes

F1=Help  ArrowKeys=Select <ENTER>=Accept <ESC>=Quit
```

4 With the cursor on `Config`, press <Enter>.

The system displays the following.

```
Install     - Install New Products
Change   - Change/Show Installed Products
Remove - Remove Installed Products
F1=Help  ArrowKeys=Select <ENTER>=Accept <ESC>=Quit
```

5 Select `Remove` and select to remove either the `Xe and Xi Family package` or the `EPC/X, C/X and Xem Family`, depending on which board was installed.

6 At the `Are You sure?` prompt, type **y**.

7 At the `Rebuild the Kernel?` prompt, type **y**.

8 After removal is successful, use the <ESC> key to exit the MPI program and return to the # prompt.

9 Type **custom** to run the custom program.

10 With `Remove` highlighted, press <Enter>.

11 Place the cursor bar on the MenuPort Interface (MPI) package line and press <Enter>.

12 Select to remove `ALL` of the MenuPort package.

13 After the removal is complete, quit the custom program.

14 To reboot the system, type **init 6**

Chapter 3: Getting started

Before using the VT100 gateway

Before you can use the VT100 gateway, you must complete the following tasks.

1 If necessary, install a multiport adapter board and expander box. See the documentation specific to your platform for further information.

2 Create the action and screen templates necessary to navigate through the host application and return the desired information (discussed in Chapter 2, "VT100 installation").

3 Under the `/u/nortel/3270` directory, create the following files, as described in this chapter:

`screen.conf`, `trs.conf`, `trs.sdf`, and `com.conf`

4 Under the `/u/nortel/vt100` directory, create the following files, as described in this chapter:

`vt100.cfg` and `vt100.ctl`

5 Before starting Symposium OPEN IVR, change the directory to the `exe` directory on the AP, and execute the `trs -c` command. This verifies that the VT100 Gateway has been installed and configured correctly, and that the screen and action templates have the appropriate syntax.

6 Start Symposium OPEN IVR on the AP. The TRS process is automatically started when you issue the `gen start` command.

If you are going to connect to the host computer via modem, you will have to configure the modem with the correct settings, attach the modem to the communication port, and dial up the host computer. Also, ensure that `DIAL_UP` is specified in the `com.conf` file, as discussed in this chapter.

Note: The installation of EXPRESS software is not required for the VT100, even though some files are shared between the VT100 and the 3270/5250.

screen.conf file

Since the VT100 protocol is character based, it has no built in mechanism for notifying the TRS that host output has ended and the application is ready for input. Creating the `screen.conf` file allows the TRS to quickly process host output by eliminating the time it spends waiting after host output ends.

The TRS uses the `screen.conf` file to determine the end of host transmission. The TRS checks every screen sent from the host against the screens you define in `screen.conf`. Each time the TRS recognizes a complete host screen, it allows the transaction to continue.

The TRS searches each host screen for one of the keywords in `screen.conf`. When it finds the keyword on the host screen, it then searches for the end of the screen. When it encounters the end of the screen host output has ended, so the TRS allows the transaction to continue. As an added checking mechanism, the TRS can also check for the beginning of the screen to ensure that it is passing along the correct screen.

`screen.conf` must reside in the `/nortel/3270` directory.

`screen.conf` should contain the definitions of every screen the TRS will encounter—one definition per line. The order of the lines does not matter since the TRS checks every screen against all the definitions in `screen.conf`. The syntax of `screen.conf` is as follows:

Keyword:Begin string:End string:

The colons (:) are used as field delimiters and must be placed in the specified positions without any additional white space.

Keyword:

The keyword identifies the screen. If the keyword contains a space or a colon, then enclose the keyword in quotation marks. Since the keyword's purpose is to identify each screen, choose a unique keyword for each different screen.

Begin string: (optional)

The TRS uses this string to determine the beginning of the screen. The begin string should be the first character or string on the screen. If it contains a space or a colon, then place the string in quotation marks. If you do not want the `screen.conf` file to check for the beginning of the screen, then enter a hyphen (-) in this field.

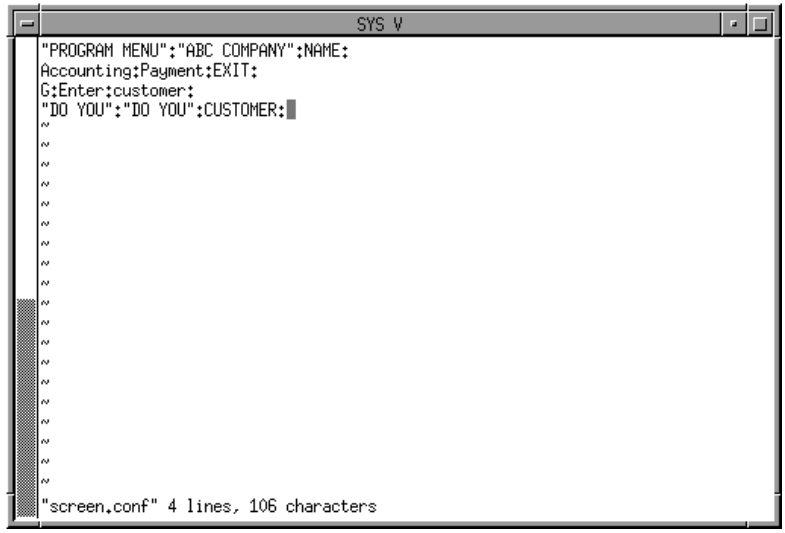
End string:

The TRS uses the end string to determine the end of the screen. The end string should be the last character or string on the screen. If the end string contains a space or a colon, then place the end string in quotation marks. You must provide an end string for the screen.conf file to be valid.

> ***Note:*** The TRS considers the begin string and the end string as part of the data.

An example of the `screen.conf` file follows.

Figure 3-1: screen.conf file



In this figure, “PROGRAM MENU” is the keyword for the first screen. Since it contains a space, it is enclosed in quotation marks. “ABC COMPANY” is the Begin String for the first screen, again in quotes because it contains a space. NAME is the End String for the first screen. When the TRS finds the screen with the keyword “ABC COMPANY”, it and begins a search for the end string NAME. In this case, it also searches for the begin string, “PROGRAM MENU”. Upon finding the begin and end string, the TRS continues the transaction.

If your host application screens do not have enough unique keywords or if the application demonstrates unpredictable screen behavior, then you will have difficulty making a `screen.conf` file. In this case, refer to the MAX_WAIT_TIME entry in the `com.conf` file discussed in this chapter.

Setting up the trs.conf file

The `trs.conf` file specifies the initial-action template for each session you define on the Symposium OPEN IVR AP. The `trs.conf` file must reside in the `/nortel/3270` directory. Figure 3-2 shows the syntax for the `trs.conf` file.

Figure 3-2: trs.conf file syntax

```
app-name:board-number>session-number:initial-template:heartbeat:protocol
```

The colons (:) and the greater than (>) symbols are used as field delimiters and must be placed in the specified positions without any additional white space.

app-name

The `app-name` entry identifies the application on the mainframe to which you assign the session. The name entered here is the same name you enter in the action templates that are executed by the VT100 Host Connectivity Configuration and Development Guide application.

board-number

The `board-number` entry must be 0. This entry is provided to support future development.

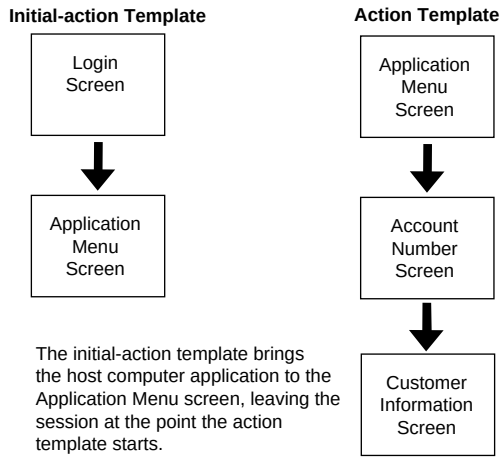
session-number

The `session-number` entry determines the number of sessions per board. Session numbers range from 2 to 32.

initial-template

The `initial-template` entry identifies an initial-action template (without the `.act` file extension) for setting the start-up action for the specified sessions when connecting to the host computer. The start-up action brings the specified sessions to the screen on the host computer where the action templates start when processing requests. Figure 3-3 shows the flow of a sample initial-action template.

Figure 3-3: Initial-action template


Initial-action Template
Login Screen
Application Menu Screen
Action Template
Application Menu Screen
Account Number Screen
Customer Information Screen
The initial-action template brings the host computer application to the Application Menu screen, leaving the session at the point the action template starts.

The initial-action template follows the same format described earlier for action templates. The screen templates specified by the initial-action template must also be created. The screen template for a login screen (or the login prompt) must specify the correct login ID and password (if appropriate) to access the host computer. You should not specify a reset-action template in the initial-action template, although you should specify a logout-action template.

heartbeat

You can specify an optional heartbeat action for the application specified by `app-name`. You can use this feature to send an indication to the host that a session is still active. Some hosts log out sessions that remain idle for a period of time. You can also use the heartbeat action to check connectivity, verifying that the session remains on the appropriate screen. Typically, the heartbeat action contains a logout-action template which brings the session back to the login screen if the connectivity with the mainframe fails. If you do not need a heartbeat action, enter a hyphen for this action.

You specify the `heartbeat` in this format: `actiontemplate@n,` where `n` is the number of seconds between each execution of the action template when the session is idle.

The `heartbeat` action template uses the same syntax as the action templates described earlier in this chapter, and usually only specifies a single screen template. That screen template is usually the last screen specified in the initial-action template. Typically, your screen template would only include a key-descriptor line, usually the <Enter> key and validation tag.

protocol

The `protocol` entry indicates the communications protocol being used by `app-name`. This field is required and should be set to `vt100`.

Example trs.conf file

Consider the following example. Initial-action template files `login.act` and `signin.act` have been defined and reside in the `/nortel/3270` directory. There are four applications which you want to access on the host computer:

- accounting - accesses the accounting software
- market - tracks stock market activity
- banking - retrieves credit balances
- airline - for purchasing tickets on a local commuter carrier

The application names shown here are not necessarily the actual names assigned on the host computer, but they are the names assigned on the AP for the `trs.conf` file and all action templates that use these applications.

You want to set up the VT100 Gateway to initialize the sessions as follows:

- Initialize sessions 2-3 for accounting with `signin.act`
- Initialize sessions 4-8 for market with `login.act`
- Initialize sessions 9-10 for banking with `login.act`

  and

- Initialize sessions 15-17 for airline with `signin.act`

Figure 3-4 illustrates how to set up `trs.conf` to implement this configuration.

Figure 3-4: Sample trs.conf file

```
#trs.conf file to set up sessions for the OPEN IVR application processor
accounting:0>2-3:signin:ping@60:vt100
market:0>4-8:login:-:vt100
banking:0>9-10:login:-:vt100
airline:0>15-17:signin:ping@60:vt100
```

Setting up the trs.sdf file

The `trs.sdf` file maps the short name of each session to the LU number.

> ***Note:*** Refer to the documentation accompanying the EXPRESS package for information about configuring EXPRESS for your environment.

Configuring EXPRESS provides you with the values for the `trs.sdf` file. The following figure displays the format for the `trs.sdf` file.

Figure 3-5: tsr.sdf file format

```
U:Username

Session Short Name    LU Number

Session Short Name    LU Number

Session Short Name    LU Number

•                     •

•                     •

•                     •
```

You can list up to eight sessions and LUs under each EXPRESS user name. The following figure shows a sample `trs.sdf` file.

Figure 3-6: Sample trs.sdf file

```
# Example trs. sdf file

 U: express1
a 2
b 3
c 4
d 5
e 6
f  7
g 8
h 9
U: express2
a 10
b 11
c 12
d 13
e 14
f 15
g 16
h 17
U: express3
a 18
b 19
“trs.sdf” 23 lines, 140 characters
```

Setting up the vt100.cfg file

The configuration process involves creating a session file that is used to define all the sessions for a particular user or group. It can be used with the EXPRESS display emulator or by entering the command `start grp`.

Defining users and session groups

The EXPRESS user and session configuration defines the EXPRESS users and the sessions and supervisory tasks available to these users. Configuration of users and sessions varies with each user population.

You define a session file for each EXPRESS user to configure the number of concurrent sessions for the user. When you start a session, a session group name is invoked. This section provides examples of session files for communication with the host.

Session configuration files

The sample session files `tn3270.cfg`, `tn5250.cfg`, and `vt100.cfg` are released in the `user_data` directory. You copy these files to the 3270 directory and use a text editor (such as `nedit`) to customize these files to define sessions. If the parameters do not change, you can reuse the files `tn3270.cfg`, `tn5250.cfg`, and `vt100.cfg` for each new EXPRESS user.

Sample session file for VT100 communications

The file that follows is an example of the session file settings for communication with VT100 host devices. Modify the template to match your environment and configuration. Name this script `vt100.cfg`, and place it in the `/u/nortel/3270` directory.

```
# Global parameters for the entire user group#

    accessPermission  Group       # UNIX permissions for session group
    :
                                  # choices: Owner/Group/All
    numUndefSess:     0           # number of dynamic sessions
                                  supported
    defaultTnHost     markets     # tn server name or address

# First VT100 session

   Session:    vt100               # VT100 session
               id:    a            # One character session id
               Name: Session_a     # Name used to identify the session
               iconName: a         # Icon name used for the session
               tnHost: markets     # telnet server name
```

```
# Second VT100 session

   Session:    vt100              # VT100 session
               id:    b           # One character session id
               Name: Session_b    # Name used to identify the session
               iconName: b        # Icon name used for the session
               tnHost: markets    # telnet server name

# Third VT100 session

   Session:    vt100              # VT100 session
               id:    c           # One character session id
               Name: Session_c    # Name used to identify the session
               iconName: c        # Icon name used for the session
               tnHost: markets    # telnet server name

# Fourth VT100 session

   Session:    vt100              # VT100 session
               id:    d           # One character session id
               Name: Session_d    # Name used to identify the session
               iconName: d        # Icon name used for the session
               tnHost: markets    # telnet server name
```

Note: The Symposium OPEN IVR TRS process handles eight sessions for each EXPRESS user. To determine the number of users to define, divide the total number of required sessions by eight. All sessions are named “a” through “h”.

Completing your configuration

Before you attempt to bring up the entire host link and run a Symposium OPEN IVR application, test each part of the host link separately. By testing parts of the host link in sequence instead of the entire configuration, you can isolate problems more easily.

Bringing up an interactive session

Bringing up an interactive session with the host is the best way to check your host connection. You see output from the host, and you can actually interact with it.

To bring up the EXPRESS display emulator, follow these steps.

1 Type the following command at the prompt:

`xtnte -sessdef` ***`session file`***

where *`session file`* is the name of the session file appropriate for the session type

2 Press <Enter>.

The Interactive Session window is displayed. This window displays all sessions:

- If you are connected to the host, this window displays the main or introductory screen for the host.
- If you are not connected to the host, the window remains empty.

Bringing up an interactive session for a particular user

To bring up an interactive session for a particular user, follow these steps.

1 Type the following command:

`xtnte -group` *`username`* `-sessdef` *`session file`*

where

`username` is the name of a user defined in the `vtekhost_adm` file

`session file` is the name of the session file appropriate for the session type

Note: This command displays a menu listing all the sessions available. Selecting a session displays an interactive screen for that session. You can use this screen

- to verity the host connection
- to cycle through the host application
- to watch the interaction between the AP software and the session

CAUTION

Risk of data loss

If you run this command when the AP software is active, do not quit `xtnte`. If you quit `xtnte`, the system terminates the connection between the client and the server.

2 To select another session, select `Open` from the File menu.

The system displays the session list again.

3 To remove the window from your screen, click the Minimize button using MB1.

Note: The minimize button is a small square located in the upper right corner of the window. Clicking this button closes the window. Its icon then appears in the Icons window.

4 To restore a window from its icon, double-click the icon using MB1.

Setting up the vt100.ctl file

The `vt100.ctl` file must be created and stored in the `/nortel/3270` directory before you can use the VT100 Gateway. The `vt100.ctl` file configures the terminal type for each session and corresponds to the sessions defined in the `trs.conf` file.

Each line in the `vt100.ctl` file uses the syntax shown in Figure 3-7.

Figure 3-7: vt100 file syntax

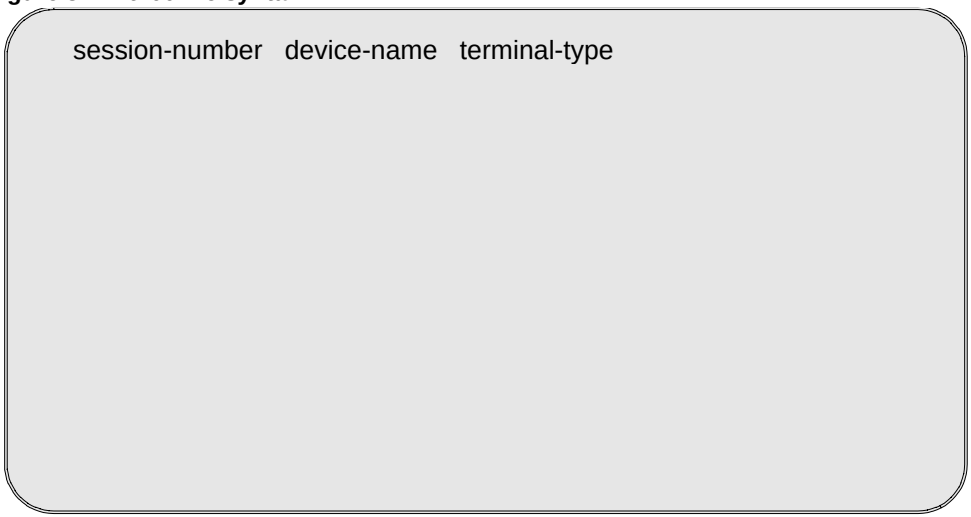


Figure 3-8 shows a `vt100.ctl` file, based on the example `trs.conf` file in Figure 3-4.

Figure 3-8: Sample vt100.ctl file

```
2  /dev/tty01 vt100
3  /dev/tty02 vt100
4  /dev/tty03 vt100
5  /dev/tty0A vt100
6  /dev/tty0B vt100
7  /dev/tty0C vt100
8  /dev/tty0D vt100
9  /dev/tty0E vt100
10 /dev/tty0F vt100
11 /dev/tty0G vt100
12 /dev/tty0H vt100
13 /dev/tty0I vt100
14 /dev/tty0J vt100
15 /dev/tty0K vt100
16 /dev/tty0L vt100
```

The following sections describe each entry in the `vt100.ctl` file.

session-number

The `session-number` entry specifies the session that this line defines. For example, the first line in Figure 3-8 defines the terminal type to be used for session 2. The entry must be a single number, ranging from 2 to 128. You do not need to list the sessions in numerical order, and you can skip session numbers.

device-name

The `device-name` entry is the exact name of the device file in the /dev directory to use for the VT100 Gateway for the specified sessions. Use the full directory path (as shown in Figure 3-8) when entering the device file name. You can set all necessary operating modes (for example, baud rate, representation parameters, flow control, and so on) in the `com.conf` file discussed in the next section.

terminal-type

The `terminal-type` entry defines the type of terminal to be emulated by the TRS process. For this release of the VT100 Gateway, this entry is always `vt100`. This field is included to support future releases that may allow additional terminal types.

Setting up the com.conf file

`com.conf` sets the communication attributes of the serial port. The `com.conf` file allows the adjustment of the terminal I/O options for the host's input device, which in this case is the Symposium OPEN IVR TRS process. If the `com.conf` file does not exist, the TRS uses its hardcoded default values. The TRS default communication settings are as follows:

Baud Rate:	9600
Parity:	None
Stop Bit:	1

The `com.conf` file must be created if the baud rate is anything other than 9600.

Creating the `com.conf` file allows the adjustment of the following values.

IXON

IXON allows you to stop host output by sending ASCII DC3, and to restart host output by sending ASCII DC1. IXON prevents input queue overflow.

IXOFF

IXOFF requests that the host send start/stop characters when the input queue is nearly empty or full. IXOFF prevents host input queue overflow.

IXON and IXOFF prevent queue overflow. If both are set and the host or terminal sends more information than their respective queues can handle, the communication port stops the data flow. When the queue gets down to a manageable level, the communication port resumes the flow of information.

Baud rate

300 to 19200. Setting the baud rate tells the TRS the rate at which it is to communicate with the host. Set the baud rate by typing **b**, then the rate. For example, B4800 or B19200.

Parity

PARENB enables parity generation and detection.

PAREVN sets parity to even.

PARODD sets parity to odd.

CSTOPB

CSTOPB selects 1 stop bit per character. If CSTOPB is not specified, then there are 2 stop bits per character.

DIAL_UP

Enter DIAL_UP in the `com.conf` file if Symposium OPEN IVR is going to access the host computer via modem.

DEBUG_LEVEL

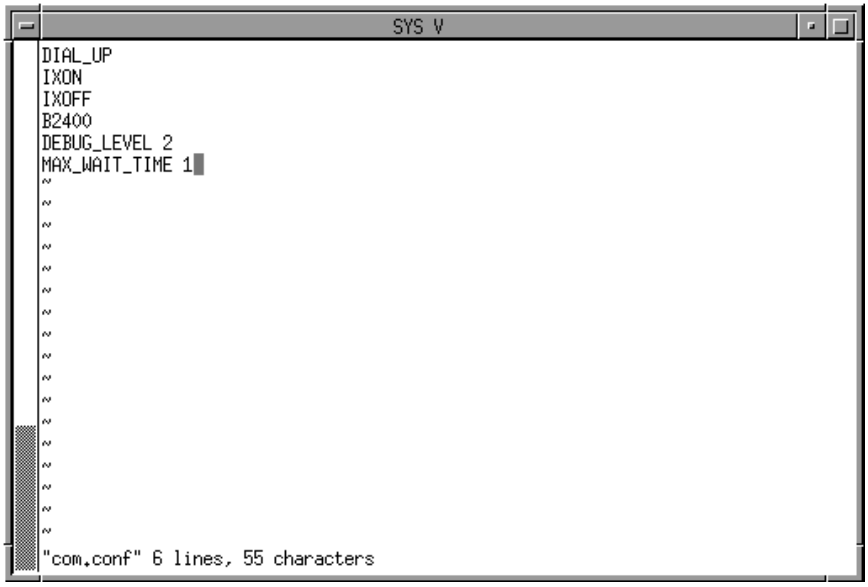
1 to 5. Setting the debug level determines how much debugging information is sent to the `vt100.log`.

MAXIMUM_WAIT_TIME

This parameter tells the TRS how long to wait for the output from the host to end. Only use this parameter if you can not use a `screen.conf` file. The wait time is specified in seconds.

The `com.conf` file should be located in the `/3270` directory. An example of a `com.conf` file follows.

Figure 3-9: Sample com.conf file



A complete sample transaction

The following sections provide an example of action and screen templates and their corresponding mainframe screens:

- initial-action template
- action template
- reset-action template
- logout-action template
- screen templates that support the above action templates

The sample transaction being performed in this section uses these templates in the following way:

- The initial-action template logs on to the host computer and starts up the “acct” application.

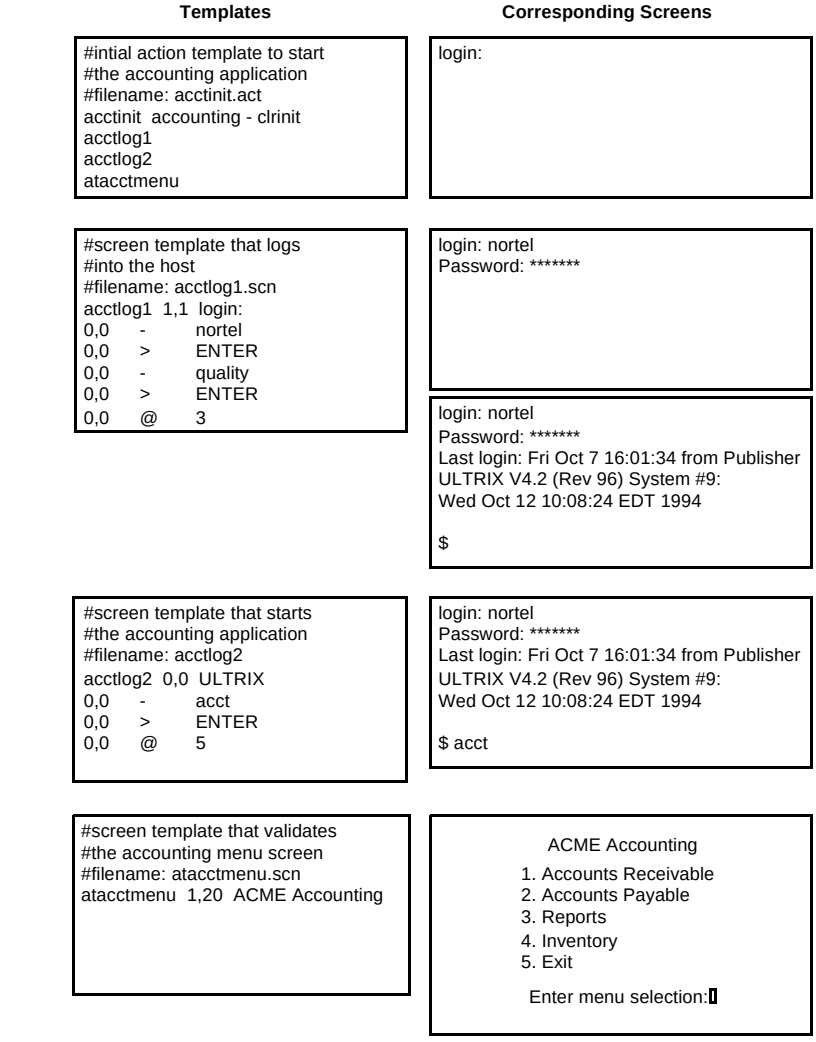
- The action template chooses the “Accounts Receivable” option from the application’s menu and retrieves a customer’s balance.
- The reset-action template returns to the application’s menu screen and waits for the next transaction.

The logout-action template is included in case an error occurs at any time during the transaction.

Initial-action template

Once the TRS process is invoked, the initial-action template `acctinit.act` brings the session assigned to the accounting application to the menu screen. Figure 3-10 shows the initial-action template, its supporting screen templates, and the corresponding screens on the host computer.

Figure 3-10: Sample initial-action template



The initial-action template starts the accounting session with the host computer by accessing accounting from the `trs.conf` file. It then executes the screen templates `acctlog1`, `acctlog2`, and then `atacctmenu`.

The screen template `acctlog1.scn` does the following:

- enters the login name

- enters the password
- waits three seconds for the host to process the login action

The screen template `acctlog2.scn` does the following:

- at the system prompt, enters the command to run the application
- types the <Enter> key to start the application
- waits five seconds for the host to run the accounting application

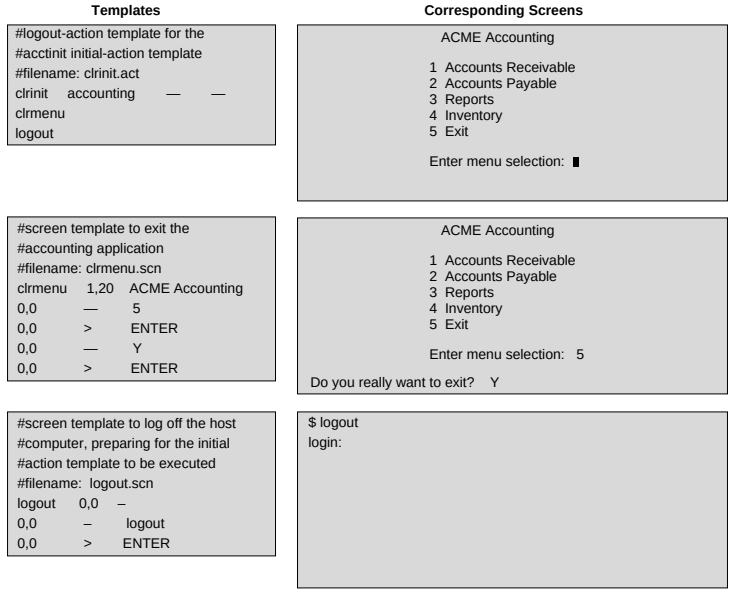
The screen template `atacctmenu.scn` validates the accounting main menu screen, but performs no action.

The `acctlog1` and `acctlog2` screen templates use "0,0" as the row, column location, and a hyphen for the field-tag to indicate that all text will be entered at the current cursor position. This method is used because the login prompt and the first prompt may not always appear in the same location on the screen. Prompts vary from system to system, and can include longer, customized names.

> ***Note:*** No key descriptor is specified for the `atacctmenu` screen template. This means that once the screen is validated, this screen remains active until the next transaction is executed.

The initial-action template shown in Figure 3-10 specifies the `clrinit.act` logout-action template, and does not specify a reset-action template. Figure 3-11 shows the `clrinit` logout-action template, its corresponding screen templates, and the associated application screens.

Figure 3-11: Sample logout-action template used by the initial-action template



The final screen template listed, `logout`, brings the host application back to the initial screen, leaving the connection open and waiting for the TRS process to log in. TRS always executes the initial-action template for a session (after a pause of 30 seconds) after any logout-action template is executed.

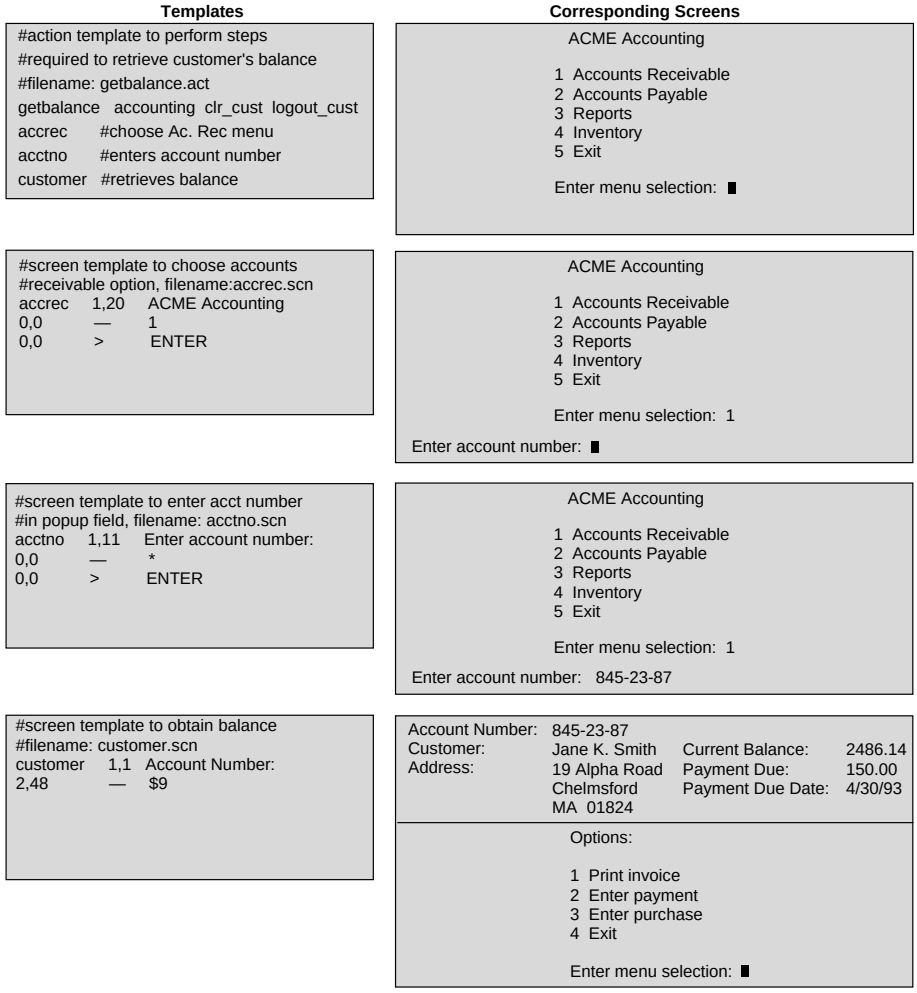
Action template performing a transaction

As shown in Figure 3-10, the initial-action template brings the session to the application's menu screen. The action template created to get a customer's balance starts at that screen. Figure 3-10 shows the action template, its screen templates, and the corresponding host computer screens that perform the following functions:

- choose the "Accounts Receivable" option from the application's menu
- enter (on the line that pops up at the bottom of the screen) the customer account number from an input buffer provided by the COMI cell

- when the customer information screen displays, the template places the current balance in an output buffer to be transmitted to the COMO cell
- execute a reset-action template to return the session application to the application's menu screen

Figure 3-12: Sample action template



The number 1 and the account number are entered at the current cursor location. Therefore, you do not need to specify a location or a field tag. The last line in the “customer” screen template places the contents of the field starting at location 2,48 (the number 2486.14) into the next output buffer.

The reset-action template is designed to return the session to the “acct” application’s menu screen where it awaits the next transaction. Figure 3-13 shows the reset-action template and its accompanying screen templates.

Figure 3-13: Sample Reset-action template

Action Template

```
#reset-action template for the getbalance
#action template, filename: clr_cust.act
clr_cust     accounting   —    —
clrcust          #exits customer info screen
atacctmenu  #validates application menu screen
```

Screen Template

```
#screen template to clear customer info
#screen, filename: clrcust.scn
clrcust       1,1    Account Number:
0,0           —      4
0,0           >      ENTER
```

```
#screen template that validates the acctng
#menu screen, filename: atacctmenu.scn
atacctmenu    1,20   ACME Accounting
```

Notice that the `atacctmenu` screen used in the reset-action template is the same screen used in the initial-action template. This screen template verifies that the menu screen has returned, but it performs no function.

Figure 3-14 shows the logout-action template and its accompanying screen templates, that return the host computer to the login screen if an error occurs and the reset-action template also experiences an error.

Figure 3-14: Sample logout-action template

Action Template

```
#logout-action template for the getbalance
#action template, filename: logout_cust.act
logout_cust     accounting    —     —
clrcust     #exits customer info screen
clrmenu  #exits applicatoin menu
logout     #logouts out to prepare for login
```

Screen Template

```
#screen template to clear customer info
#screen, filename: clrcust.scn
clrcust       1,1     Account Number:
0,0            —      4
0,0            >      ENTER
```

```
#screen template to exit the
#accounting application
#filename: clrmenu.scn
clrmenu     1,20    ACME Accounting
0,0           —       5
0,0           >       ENTER
0,0           —       Y
0,0           >       ENTER
```

```
#screen template to log off the host
#computer, preparing for the initial
#action template to be executed
#filename:  logout.scn
logout      0,0     –
0,0           –       logout
0,0           >       ENTER
```

The logout-action template brings the host computer back to the login screen. The TRS then executes the initial-action template, as it does whenever any logout-action template is successful.

If the transaction fails before the customer screen is accessed, the logout-action template searches the other screen templates until a valid tag match is found, then it executes that screen template and the following screen templates.

Chapter 4: Template files

The TRS process uses action and screen templates to maneuver through the screens of a host application. These templates exchange information with the host application screens and transfer information to and from the TRS's buffers. Coupled with the VT100 emulation software and hardware, they provide the host with exactly the same type of input as a terminal operator.

This chapter explains how to do the following:

- Determine the actions a terminal operator performs to enter and retrieve information.
- Create the action and initial action template files that define the sequence of host application screens for each transaction.
- Create the screen template files that define the sequence of fields encountered on each screen.

  Note: If you make backups of your template files, do not store them in the `u/nortel/3270` directory, or in any subdirectory under `/3270`. You should make a directory at the same hierarchical level or higher as `/3270` (for example, `/nortel/3270bak`). The TRS process searches the `/3270` directory and any subdirectories within it for files with the `.act` or `.scn` extensions.

Determining the required transactions

Imagine that you are an operator sitting at a terminal. In order to perform a specific task, you type information and press function keys until you accomplish the desired task. Perhaps a caller asks you to look up a customer's account balance or enter an order. The series of steps you perform at the terminal enable the application on the host computer to complete the transaction.

To develop a voice application that accesses the same information as a terminal operator, you need to tell Symposium OPEN IVR how to execute the same series of actions that the terminal operator executes. You provide this information in ASCII files called template files.

Template files provide the layout and content of each screen in the host application as the terminal operator sees them. Figure 4-1 compares a transaction done by a terminal operator to one done by a customer calling in to a voice response system.

> ***Note:*** The first step performed by the terminal operator is not performed by the action template. It is performed by the initial-action template (described later in this chapter). The initial-action template handles the login and moves the application to the appropriate screen to begin the transaction.

Figure 4-1: Sample transaction—terminal operator versus Interactive Voice Response system

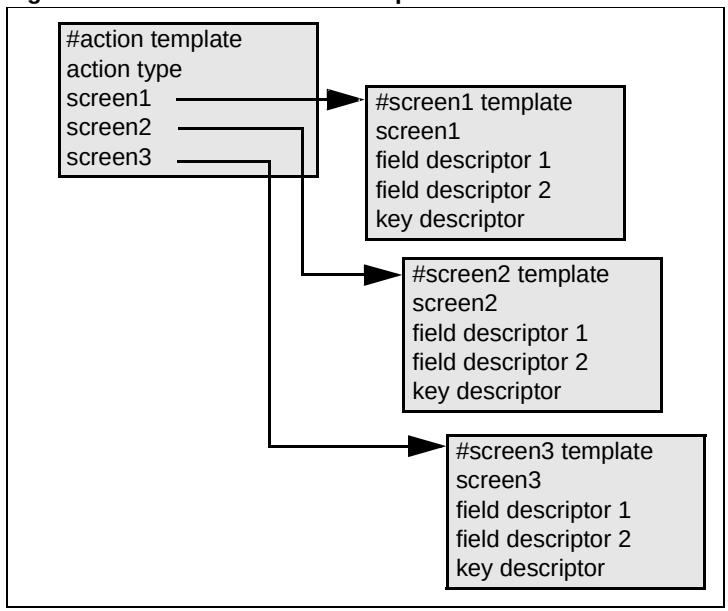
An operator follows this sequence to retrieve data:	**A customer follows this sequence to retrieve data:**
1 Starts the accounting application.	1 Calls into the MRS, and activates a voice application.
2 Selects the Accounts Receivable menu option.	2 When prompted, selects the Accounts Receivable option from a menu prompt.
3 Asks the caller for account information.	3 When prompted, enters account information.
4 Enters the caller's account number.	4 At this point, the application sends a request for information to the TRS. The TRS then executes the Action template for this specific transaction.
5 Reports the balance when the customer's information appears on the screen.	5 Hears the account balance information played back.

The TRS requires two types of template files:

- *Action templates* describe the sequence of screens traversed in order to perform a specific transaction. Figure 4-1 includes a sample action template file.
- *Screen templates* validate each screen, define the fields on the screen that require data, and define all keystrokes required for the screen. In Figure 4-1, `acctrec`, `acctno`, and `customer` (shown in the action template file) are screen templates.

Figure 4-2 shows how screen templates relate to action templates.

Figure 4-2: Action and Screen templates



VT100-based applications often have format inconsistencies that make it difficult for the TRS to efficiently determine when the host application is ready for input and what region of the host screen contains vital information. These inconsistencies are due to the character-based nature of the VT100 protocol.

In addition, the VT100 communication protocol has no way of notifying the TRS that host data transmission has ended. From the terminal operator's perspective, it is easy to tell when host transmission ends because the operator's requested information appears on the screen. From the TRS's perspective, there is nothing inherent in the VT100 protocol to provide notification of the end of host output. The TRS encounters a stream of data from the host and, from this, must determine the identity of each screen as well as locate the end each screen. To enable the TRS to do this, as well as to cope with screen inconsistencies, you must create a file called `screen.conf`.

The action templates, screen templates, and `screen.conf` file are ASCII text files that use a simple syntax to define the screen flow and input/output fields. The sections that follow provide a detailed explanation of the templates, as well as the information necessary to create the `screen.conf` file.

Action templates

A VT100 transaction typically moves through several screens until it locates specific information. The screens may be a series of commands issued at the operating system prompt, or they may be screens within an application running on the host computer. Whenever Symposium OPEN IVR references an action template, the VT100 Gateway executes the screen templates listed in the action template, moving through the application just as a terminal operator does. An action template must specify the same sequence of screens that the terminal operator traverses.

A separate action template defines each transaction. In the example shown in Figure 4-1, if you want to select a menu option other than "Accounts Receivable," you define another action template.

Action templates describe the flow of the screens that comprise a particular transaction. For example, if you want a transaction to access billing information for a specific client, as a terminal operator you perform the following steps.

1 Log in to the computer.

2 Start the "acct" application.

3 Select the "Accounts Receivable" menu.

4 When prompted, enter the client's account number and press the <Enter> key. A screen appears displaying the client's billing information.

5 Read the account information on the screen.

In a Symposium OPEN IVR VT100 transaction, an initial-action template performs steps 1 and 2. The initial-action template automatically executes when the TRS process starts (initial-action templates are described later in this chapter.). An action template created to execute this transaction performs steps 3 through 5.

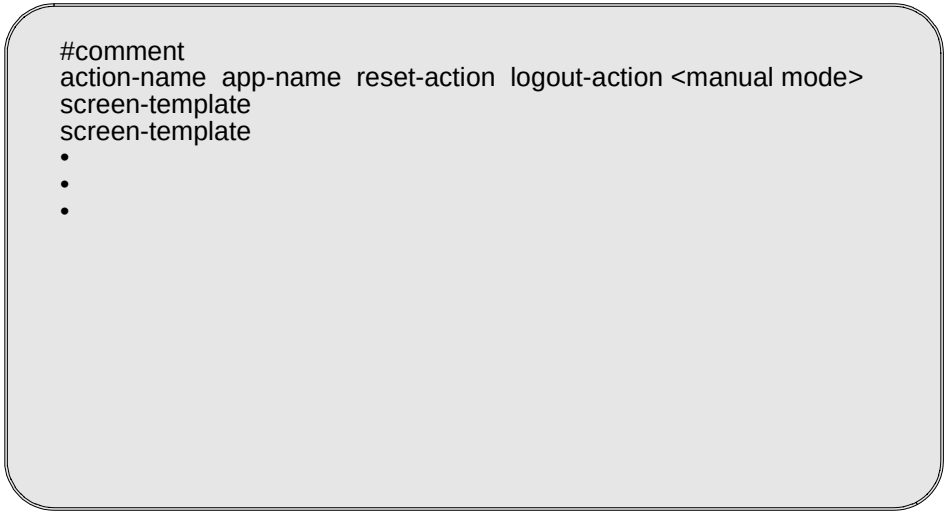
Action template syntax

An action template is an ASCII file created with a text editor. The action template files you create must reside in the `u/nortel/3270` directory or in a subdirectory below `u/nortel/3270`, and must have the file name extension `.act`. For example, if you create an action template called `getbalance.act`, it has this path:

```
u/nortel/3270/getbalance.act
```

The syntax of an action template is shown in Figure 4-3.

Figure 4-3: Action template syntax



The lines depicted as • are additional screen templates used in the transaction. Each screen template corresponds to a specific host application screen, and is listed in the action template in the same order as encountered during an actual terminal session (screen templates are discussed later in this chapter). The example in Figure 4-4 illustrates an action template file which describes a transaction for retrieving account information.

Figure 4-4: Action template example

```
#Example action template file: filename is getbalance.act
getbalance  accounting  reset_cust  logout_cust
#acctrec chooses the billing option from the main menu
acctrec
#acctno specifies the account number for the customer
acctno
#customer displays the customer information
customer
```

In Figure 4-4, the action-name is `getbalance`, the name of the action template file without the `.act` extension. The app-name is `accounting`. The reset-action is `reset_cust` (file name `reset_cust.act`), and the logout-action is `logout_cust` (file name `logout_cust.act`). Manual mode is omitted because manual mode is not required for this transaction (a description of manual mode follows on page 4-12).

The remaining lines identify the sequence of screens (`acctrec`, `acctno`, and `customer`) that the TRS must traverse to retrieve the customer billing information. These screens are listed in the order in which they must be accessed.

An explanation of each entry in the action template syntax follows.

#comment

The first line of the template in Figure 4-3 is a comment. The comment line is not required but is recommended to describe the purpose of the action template.

Comments start with the "#" symbol and can be embedded anywhere in the action template. If a comment begins a line, no non-comment fields may follow the comment in that line. However, a comment may appear after a non-comment field, such as after the screen template name as shown in Figure 4-4. It is good practice to heavily comment files so that changes can be made easily in the future.

action-name

The action-name is the file name of the action template file without the `.act` extension. The action-name is required to begin the transaction. For example, if the action template's file name is `getbalance.act`, enter `getbalance` for the action-name.

app-name

When you set up your AP with the VT100 Gateway, you must create a `trs.conf` file that assigns TRS session numbers to the application on the host computer (the `trs.conf` file is described in Chapter 3, "Getting started"). Choose a name for the host computer application name; it does not need to match any actual application name on the host computer.

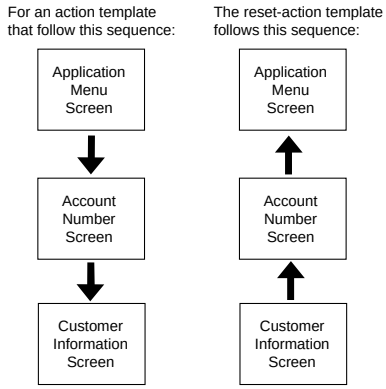
As an example, this guide uses the app-name `accounting` to represent the host computer "acct" application.

The app-name you specify in the action template must exist in the `trs.conf` file (discussed in Chapter 3, "Getting started"). Symposium OPEN IVR passes the app-name to the TRS function, which uses the name to start the appropriate session with the host computer.

reset-action

The reset-action specifies an action template to be processed when the transaction completes or if the transaction fails. Typically, the reset-action template is used to bring the host computer application back to its main screen so it is ready to process the same type of transaction. Figure 4-5 shows the sequence a sample reset-action template follows.

Figure 4-5: Sample reset-action template sequence


For an action template that follow this sequence:
Application Menu Screen
Account Number Screen
Customer Information Screen
The reset-action template follows this sequence:
Application Menu Screen
Account Number Screen
Customer Information Screen

Entering a hyphen (–) in the reset-action entry indicates that no reset-action template is specified.

If no reset-action template is specified and the transaction being executed by the action template fails, the logout-action template (described in the next section) is executed. If the transaction succeeds and there is no reset-action template specified, the host computer application remains at the screen where the transaction ended.

When you create a reset-action template, do not specify reset-action or logout-action templates in it. For example, Figure 4-6 shows a sample reset-action template.

Figure 4-6: Sample reset-action template

```
#This reset-action template returns the host computer application
#to the main menu screen from the customer information screen
#filename: reset_cust.act
reset_cust  accounting  –  –
clrcust
#exit the customer information screen
atmenu
#leave the session at the menu screen
```

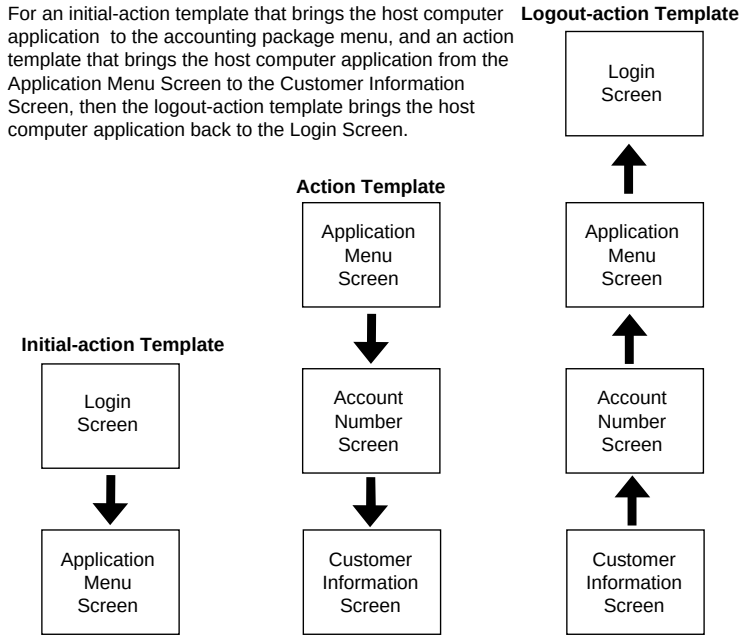
The action template using this reset-action template enters `reset_cust` as the reset-action entry.

logout-action

The logout-action specifies a logout-action template to be executed if the reset-action template fails, or if the transaction fails and there is no reset-action template specified. If the transaction succeeds, the logout-action template is not executed.

The TRS uses the logout-action template to return the failed transaction to the initial screen (usually a login screen). After it successfully executes the logout-action template, it executes the initial-action template (described on page 4-26) after 30 seconds. Figure 4-7 shows the flow of the logout-action template.

Figure 4-7: Logout-action flow


For an initial-action template that brings the host computer application to the accounting package menu, and an action template that brings the host computer application from the Application Menu Screen to the Customer Information Screen, then the logout-action template brings the host computer application back to the Login Screen.
Logout-action Template
Login Screen
Action Template
Application Menu Screen
Application Menu Screen
Initial-action Template
Login Screen
Account Number Screen
Account Number Screen
Application Menu Screen
Customer Information Screen
Customer Information Screen

The logout-action template locates the screen where the transaction failed. If, for example, the transaction failed at the account number screen, the logout-action template locates the screen template with the appropriate validation tag and starts from that screen.

Entering a hyphen (–) for the logout-action entry indicates that no logout-action template is specified. If neither a reset-action template nor a logout-action template is specified and the transaction fails, the host computer application remains at the point where the transaction failed. Future transactions that try to use this session would also experience errors because the screen where the session remained would not be the expected starting screen (unless the transaction could start from any screen).

When you create a logout-action template, do not specify reset-action or logout-action templates. Figure 4-8 shows a sample logout-action template.

Figure 4-8: Sample Logout-action template

```
#This logout-action template returns the application to the login
#screen from the customer information screen
logout_cust  accounting  –  –
clrcust
#exits the customer information screen
clrmenu
#exits the acct application, shows the system prompt
```

screen-template

The screen-template (the file name of the template without the `.scn` extension) identifies the screen template used during the VT100 transaction. Enter the screen templates in the exact order in which they appear during the transaction. Each screen template must be listed on a separate line. The syntax for screen templates is described later in this chapter.

<manual mode> (optional field)

The <manual-mode> entry allows you to attach a session resource to a particular channel. You can then use the same session for consecutive Symposium OPEN IVR transactions. This type of session is not released when the transaction is finished. To exit manual mode, you must execute a COMA cell in the Symposium OPEN IVR application or process another action template that does not contain the manual mode option. Chapter 5, “Symposium OPEN IVR call flow interface”, describes how to use the COMA cell.

To specify manual mode, type **`manual`** after the logout-action template name. If you omit manual, automatic mode is used for the template. In automatic mode, the session assigned to the action template is free for use when the transaction is completed.

> ***Note:*** You should not specify a reset-action template in an action template that uses manual mode. Manual mode is designed to stay at a specific screen. The next transaction received by the session will start at the last screen of the action template that used manual mode. This next transaction should use automatic mode and specify a reset-action template that brings the session back to the starting screen of the first action template (that specified manual mode).

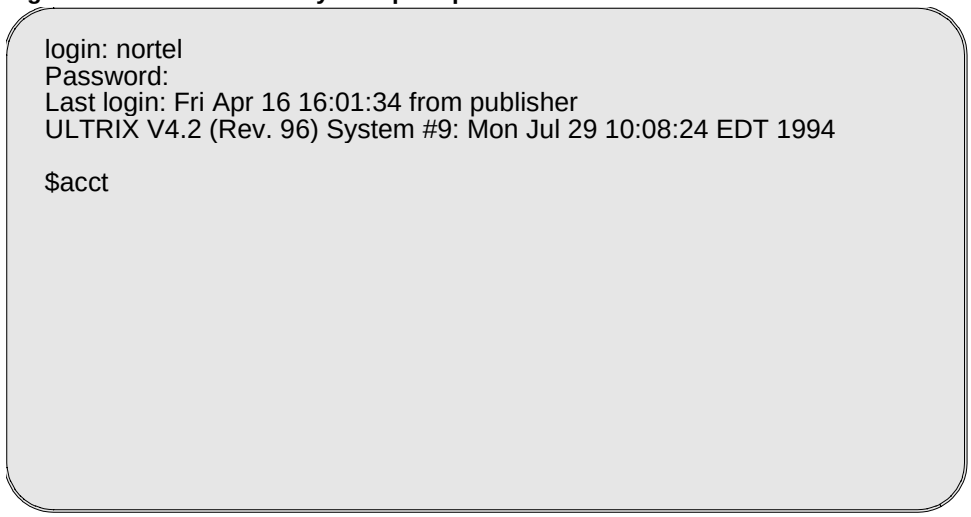
Screen templates

Screens used by the host computer could be a series of commands entered at the system prompt coupled with the system's response to those commands, or screens defined by applications on the host computer. You should define screen templates that issue the system commands to start the application (usually as part of the initial-action template), and then screen templates that make menu selections and enter or retrieve data from the screens displayed by the application. This guide uses an accounting application as an example.

Each screen contains fields. For the VT100 Gateway, a field is any place on the screen where data is entered or displayed. For example, the cursor location after a system prompt where you type a command is considered a field. Also, within an application, the area on the screen where an account balance is displayed is also considered a field (the traditional definition of a field).

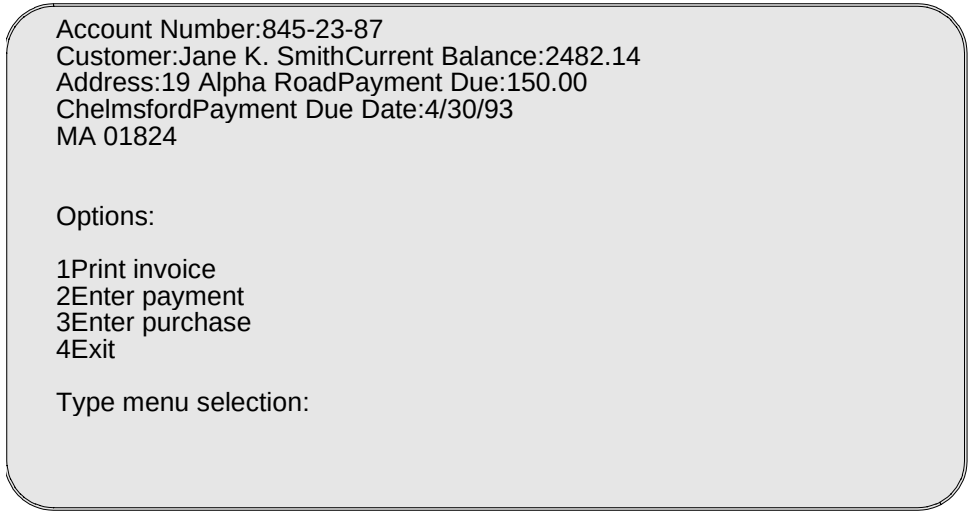
For a specific transaction, specific data is entered in certain fields, and data is read from other fields. A screen template identifies those fields on the screen that are used to process a transaction. Only the fields that are used in the transaction are included in the template. Figure 4-9 shows a sample screen.

Figure 4-9: Fields and the system prompt


login: nortel
Password:
Last login: Fri Apr 16 16:01:34 from publisher
ULTRIX V4.2 (Rev. 96) System #9: Mon Jul 29 10:08:24 EDT 1994
$acct

The cursor is shown in Figure 4-9 because if the <Enter> key is pressed, the application starts and the screen is replaced by the application screen. In Figure 4-9, “nortel” is entered into the *login* field. A sample application screen showing fields is shown in Figure 4-10.

Figure 4-10: Fields in a host computer application



Note: The screens shown in this guide are examples and are not part of any real application.

In Figure 4-10, “Jane K. Smith” is entered in a field.

Different transactions may access different fields on a screen. For example, a transaction to locate the payment due only needs to access that field, while a transaction retrieving the customer’s balance only needs to access the *Current Balance* field.

A screen template is an ASCII file created with a text editor. The screen template files you create must reside in or under the Symposium OPEN IVR `u/nortel/3270` directory, and must have the file name extension `.scn`. For example, if you create a screen template called `customer.scn`, it has this path:

```
/u/nortel/3270/customer.scn
```

For each accessed field, there should be a field descriptor specified that governs how data is retrieved from or entered in the field. The screen template can include both data input entries as well as data output entries.

Screen template syntax

The syntax of a screen template is shown in Figure 4-11.

Figure 4-11: Screen template syntax

```
#comment
screen-name   validation tag offset   validation tag
field-descriptor
field-descriptor
  •
  •
  •
key-descriptor
sleep-descriptor
```

The lines depicted as • are additional field-descriptor lines. The example in Figure 4-12 illustrates a screen template file that obtains the balance from the screen shown in Figure 4-10.

Figure 4-12: Screen template example

```
#Screen template file to obtain the current balance; filename: customer.scn
customer1,1Account
#places balance into buffer
0,0Balance:$
```

In Figure 4-12, the first line is a comment describing the screen template file. The screen-name is “customer”, the name of the screen template file without the `.scn` extension. The “1,1” represents the location of the validation tag on the screen. The row is listed first, followed by the column. “Account” is the screen validation tag.

The third line is the field descriptor that describes an action to take. This field descriptor is going to find an exact match to “Balance:” and place the contents of the field into a buffer. The field-descriptor line has many variations, depending on what you want to do with a field. For example, the third line in Figure 4-12 could be entered as:

```
2,48– $
```

This places the contents of the field starting at 2,48 into the next buffer. (See “field-I/O” on page 4-22 for more information.)

An explanation of each entry in the screen template follows.

#comment

The first line of the template in Figure 4-12 is a comment. The comment line is not required but is recommended to describe the purpose of the screen template.

Comments can be embedded anywhere in the screen template and start with the "#" symbol. If a comment begins a line, no non-comment fields may follow the comment. However, a comment may appear after a non-comment field, such as after the field descriptor line shown in Figure 4-12. It is good programming practice to heavily comment files.

screen-name

The first non-comment line specifies the name of the screen. This is the screen template file name without the `.scn` extension.

validation-tag offset

This entry specifies the position of the validation-tag by row and column. You may enter 0,0 for the validation-tag offset in only two cases:

1 To indicate that you do not want to validate this screen (you also need to enter a hyphen (–) as the validation tag). You should only ignore the identity of a screen if you want the TRS process to perform the actions specified in the screen template regardless of what screen is actually active. For example, if you want to execute a command at the system prompt, you do not need to verify that you are at a specific screen, as long as you are sure the screen has a system prompt.

Note: In general, you should validate screens whenever possible. This ensures that data for the host computer application is sent to the correct screen, and data sent back to the Symposium OPEN IVR application is from the correct screen.

2 To tell the TRS to search the screen for the validation tag. If you do not know the exact location of the validation tag or, if the location of the validation varies, you can tell the TRS to search the screen for the tag. Enter 0,0 for the offset, then enter the validation tag in the appropriate field.

validation-tag

This entry specifies the validation tag used on the screen. The entry should be text that always appears in the same location every time this screen displays. For the example, “Account” is always displayed starting at location 1,1 whenever a customer’s Account Receivable screen is displayed.

At the end of the previous screen template, you may want to use a clear screen function (or execute the <Enter> key 24 times using key-descriptor lines) so you know the starting point for information on the screen, especially if the screens that you are accessing on the host computer scroll. For example, if you execute an application and it starts displaying information at the current cursor location, you need to know the current cursor location to be able to validate that screen.

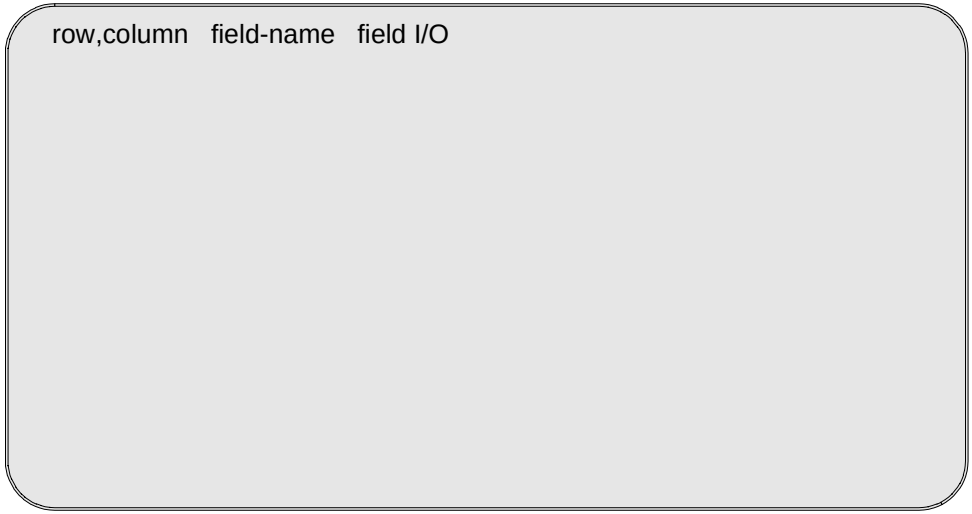
Enter a hyphen (–) for the validation-tag to indicate that you do not want to validate this screen (you would also need to enter 0,0 as the validation-tag offset). As described previously, you should only use this method of validation if you want the TRS process to perform the action specified in the template regardless of what screen is actually active.

> ***Note:*** If the screen you need to validate is a blank screen, enter this line for the items screen-name, validation-tag offset, and validation-tag:
>
> blank1,1 BLANKSCREEN
>
> The word “blank” is a place-holder; the 1,1 and BLANKSCREEN are required.

field-descriptor

This line identifies the location and name of a field on the screen, and the action to be performed. You can enter as many field-descriptor lines as necessary to perform the task needed on the screen. The field-descriptor lines should be entered in the same order as they are accessed for the transaction. Figure 4-13 shows the syntax for field-descriptor lines.

Figure 4-13: field-descriptor syntax



Note: If the application running on the host is stream-based, meaning the screen scrolls as the user enters data and retrieves responses, enter the field-descriptor as follows:

```
0,0 -  BLANKOUT
```

In this instance, the field descriptor clears the TRS memory space associated with the application screen so that your application call flow knows where to retrieve the appropriate data.

row,column

If the TRS is going to read information from a field on the host screen, this entry locates the field on the screen. If you know the exact location of the field the TRS is reading from, you can enter it in row,column format. If you do not know the exact location of the field from which the TRS is reading or, if the location of the field varies, you can enter 0,0 and the TRS will search for a match to the name specified in field-name. For unformatted, character-based applications, you must specify the exact location.

In VT100 transactions, writing to the terminal screen always occurs at the current cursor position. Therefore, if the field I/O action is writing text to the host screen, the TRS will write the text to the screen at the current cursor position regardless of the row,column you specify.

If you enter "0,0" and a hyphen for the row,column and field-name entries, the field I/O specified is executed at the current cursor location.

To locate field-names on the screen for reading, the offset of the upper left corner of the screen is 1,1 and the lower right corner is 24,80.

field-name

This entry identifies the field being accessed on the terminal screen. If the field-descriptor is 0,0 (that is, for an exact match), the field-name must uniquely match text on the screen or be a hyphen to indicate that the field is at the current cursor position. When entering the field name, keep in mind that the TRS process right-justifies all field names and removes all extra white space. If the transaction fails, this field identifies the current screen so the reset-action template can be executed.

If you enter "0,0" and a hyphen for the row,column and field-name entries, the field I/O specified is executed at the current cursor location.

To specify a specific location on the screen that does not have an associated field-name, use a hyphen (–) for the field-name entry. If you do, you must specify a location such as 24,8 for the location of the action specified by the field-I/O entry.

field-I/O

This entry indicates the action to be taken on the field. Table 4-1 explains the valid entries for this field.

Table 4-1: Valid field I/O entries

Entry	Description
*	Inputs the contents of the next input buffer (transmitted from the Symposium OPEN IVR application) into the field.
$num	Outputs the contents of the field into the next output buffer. num is a number in the range 1–31 and represents the number of characters in the field that TRS will put in the output buffer. If you do not assign num a value, TRS will place 31 characters into the buffer.
%n$	Outputs the contents of the field into an internal variable named *n*, which must be a number from 1–9.
%n*	Enters the contents of internal variable n into the field.
%n$num	Outputs the first num characters of the field into variable *n.*
text	Any text string to be entered in the field. If any of the special characters listed in this table are entered as text, enclose the entire text in quotes (for example, “$abc”).
BLANKOUT	Clears TRS memory space associated with the host application screen before retrieving output.

Note: Place the asterisk and dollar sign characters in quotation marks if your field descriptors require their use without their associated buffer commands.

For retrieving data into an output buffer, `$num` indicates the number of characters to be retrieved from the field; using `$` without a number retrieves characters until an attribute is encountered, or a maximum of 31 characters have been retrieved.

Internal variables (indicated by the % symbol) are used within the VT100 screens only and cannot be transmitted through the TRS gateway. You can only use internal variables to store and enter data from one screen to another in the host computer application. To send the data to the Symposium OPEN IVR application that called the TRS process, you need to store the data in an output buffer.

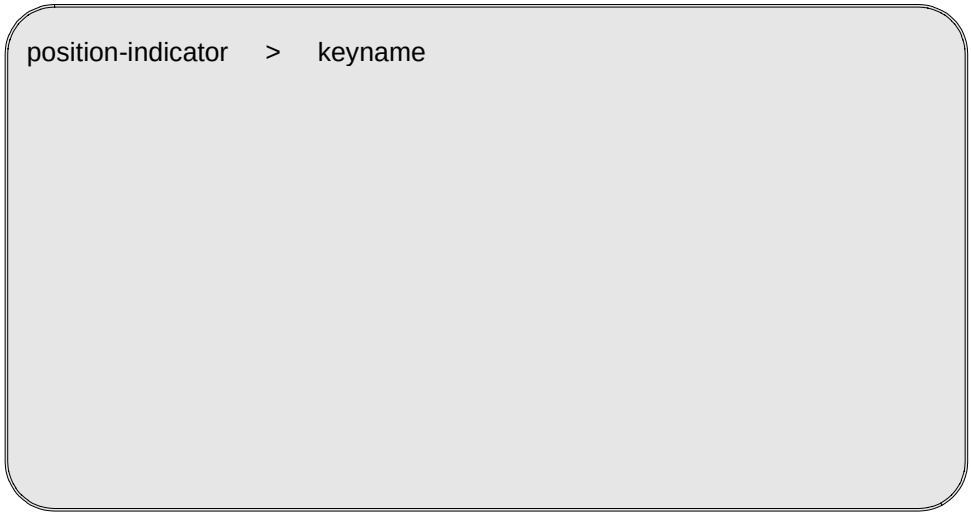
When entering a field descriptor, the entries on the line should be separated by white space or tab characters. If you are entering text for the field I/O, the TRS ignores any white space you include with the value until it encounters the new line character (for example, the value includes several words).

> ***Note:*** The order of the field-descriptor lines in your screen template file determines how data is entered and retrieved from the screens and written into the input and output buffers of a COMO cell. Chapter 5, "Symposium OPEN IVR call flow interface", describes how to use Symposium OPEN IVR cells to retrieve data from the host computer.

key-descriptor

Identifies a key to be used with the screen. To send information to the computer for processing or to execute a command, the terminal operator presses a key on the terminal keyboard. If you do not specify a key, nothing is sent to the host and the current screen does not change. The key can be the <Enter> key or a function key. The format of this line is similar to the field descriptor. Figure 4-14 contains the format of the key descriptor.

Figure 4-14: Format of key-descriptor line



position-indicator

This entry should be set to 0,0.

>

This character indicates that this line contains a key.

keyname

The name of the key for this screen. Table 4-2 lists the valid keys you can enter.

Table 4-2: Valid key names

ATTENTION	FORWARDWORD	PF11
BACKSPACE	HOME	PF12
BACKTAB	INSERTCHAR	PF13
BACKWWORD	NEWLINE	PF14
CLEAR	PA1	PF15

CURSORDOWN	PA2	PF16
CURSORLEFT	PA3	PF17
CURSORLEFTDBL	PF1	PF18
CURSORRIGHT	PF2	PF19
CURSORRIGHTDBL	PF3	PF20
CURSORUP	PF4	PF21
DELCHAR	PF5	PF22
DUP	PF6	PF23
ENTER	PF7	PF24
ERASEEOF	PF8	RESET
ERASEINPUT	PF9	SYSREQUEST
FIELDMARK	PF10	TAB

This line may appear anywhere after the screen-name line (that is, the first non-comment line). As with field-descriptor lines, the entries on this line must be separated by white space or tab characters.

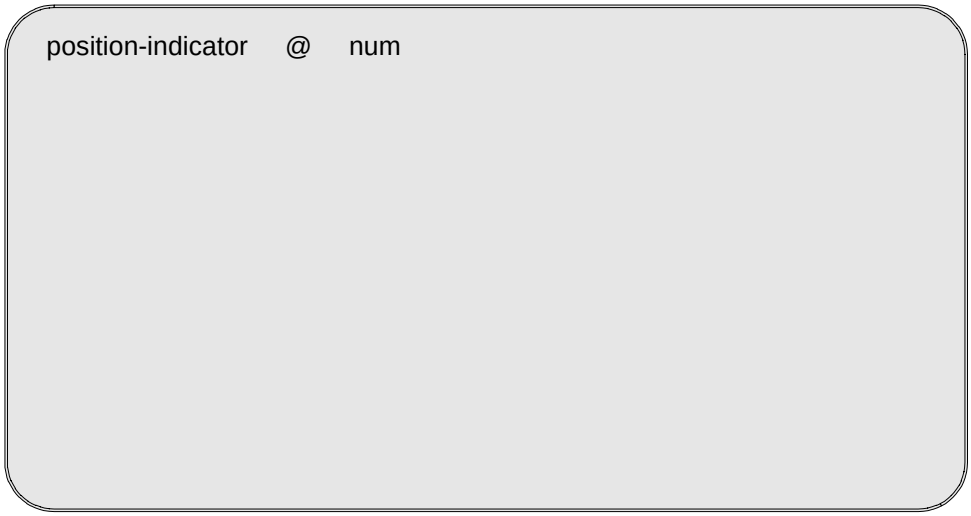
sleep-descriptor (optional)

A sleep descriptor causes the transaction to pause for a specified number of seconds. You can use a sleep descriptor to pause the transaction for a specified period of time before or after processing a key descriptor. The waiting period takes into account the time the host computer may take to process the information entered on the screen, and to move to the next screen. The transaction waits at any point where you place a sleep descriptor.

Usually, a sleep descriptor is placed after a key descriptor to indicate that the session should wait for a specified amount of time to ensure that the next screen is ready.

Each sleep descriptor follows the syntax shown in Figure 4-15.

Figure 4-15: Sleep-descriptor format



position-indicator

This entry should be set to 0,0.

@

Identifies that this line is a sleep descriptor.

num

Specifies the number of seconds that the session should sleep. For example, to wait 25 seconds, code the sleep descriptor as

```
0,0 @ 25
```

Initial-action templates

Before you can process any information on the host computer using the VT100 Gateway, you need to set the starting point for each of your terminal sessions. For example, you may want session 2 to start processing at an application's main menu, whereas session 5 should start at the system prompt.

You automatically initialize each terminal session by defining initial-action templates. Initial-action templates are action templates that specify a sequence of screen templates that position the terminal session at the desired location on the host computer whenever TRS is started up.

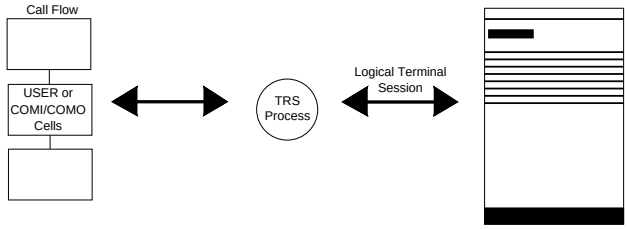
The initial-action template follows the same format and syntax as defined for action templates earlier in this chapter. If you must use "*" or "$" characters in your field-descriptors, put them in quotation marks.

Instead of referencing these action templates in the COMI cell, you specify initial-action templates in an ASCII file named `trs.conf`. In addition, the session numbers defined in the `trs.conf` file must have a corresponding entry in a file named `vt100.ctl`, which assigns a device type for the session. See Chapter 3, "Getting started", for information on the configuration and syntax of the `trs.conf` and `vt100.ctl` files.

Chapter 5: Symposium OPEN IVR call flow interface

Now that you understand how to script VT100 transactions as action and screen templates, you can begin integrating these templates into your Symposium OPEN IVR Applications (see Figure 5-1).

Figure 5-1: Accessing the host computer



This chapter explains the following:

- how to use the COMI, COMO, and COMA cells to communicate with the host computer
- how to code your applications to perform VT100 transactions

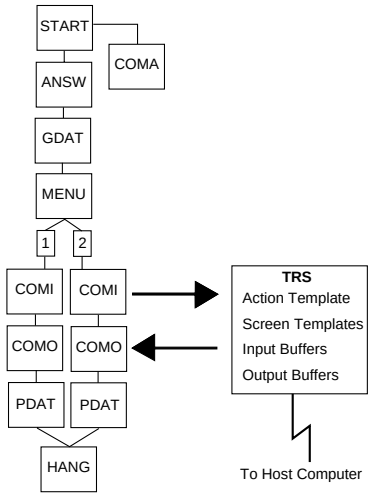
Using the COMI, COMO, and COMA cells to access the host computer

With Symposium OPEN IVR, the links between the Symposium OPEN IVR call flow and the TRS are the COM cells:

- The COMI cell sends input to the TRS process and identifies the action template file.
- The COMO cell receives output from the TRS process.
- The COMA cell aborts a transaction in progress (usually used in the clean-up branch in case a caller hangs up before the transaction completes).

Figure 5-2 provides an illustration.

Figure 5-2: Activating the gateway from a COMI cell



Developing an application that processes terminal sessions

To develop an application that processes one or more terminal sessions, simply start as if you were developing any other voice application:

- Determine the telephone interaction with the caller and create the corresponding Symposium OPEN IVR call flow.
- At the point in the call flow where you require interaction with the host computer, insert a COMI cell.
- Insert a COMO cell to receive the output from the TRS transaction.

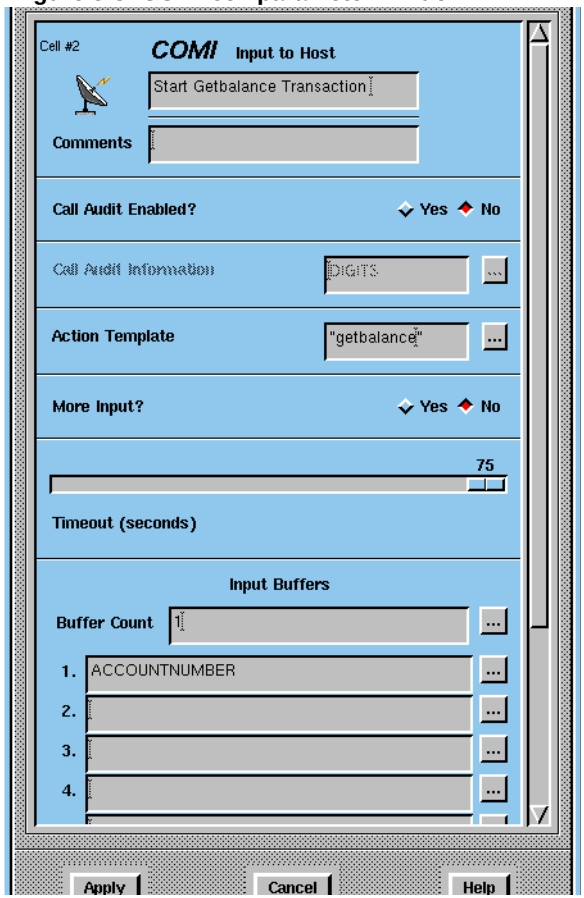
You must always follow a COMI cell with a COMO cell, even if you do not require the return of any data.

The following sections show you how to code each of these cells for use with the same example transaction presented in Chapter 3, “Getting started”.

Setting the COMI cell parameters

Figure 5-3 shows the parameter window for the COMI cell.

Figure 5-3: COMI cell parameter window



The following sections describe what you should enter in each area of the parameter window.

COMI cell name

In order to make the cell easy to identify, include the name of the action template that the cell calls out. The cell name shown in Figure 5-3 is “start getbalance transaction.”

COMI action template

Enter the name of the action template, in quotes, and leave off the `.act` suffix. Figure 5-3 shows “getbalance” as the entered name.

COMI cell More Input

If you need to send more than 10 input buffers to the host computer, click the button in front of “Yes”; if you are sending 10 or less input buffers, click the button in front of “No”. If you specify “Yes”, place additional COMI cells on the drawing board until you have enough input buffers. Connect the Success branch of the first COMI cell to the Input branch of the following COMI cell. Connect the rest of the COMI cells in this fashion. The final COMI cell should have the “No” button selected for the “More Input?” parameter.

COMI cell Timeout

Select the number of seconds you want the TRS process to wait for the transaction to complete by moving the slider under “Timeout”. There is an available range from 1 to 75 seconds. This indicates the maximum time for the COMI cell to start the transaction and send the necessary input buffers, and for the COMO cell to receive the output buffers. The TRS process rejects the transaction and sends back a timeout if the action is not performed within the time allotted.

> ***Note:*** The application will take the timeout branch of the COMI cell if a timeout occurs.

COMI cell Buffer Count

You do not need to enter a value in this field as Symposium OPEN IVR automatically calculates the number of buffers when you click the Apply button.

COMI cell Input Buffers

Enter the names of the input buffers containing the information to pass to the TRS process. The input buffer shown in Figure 5-3 is ACCOUNTNUMBER, without any quotes. In this example, you must program your application to place the customer's account number in the ACCOUNTNUMBER buffer before the COMI cell is executed.

In this example, ACCOUNTNUMBER is a user-defined buffer; you can use any name for your applications, or you can use system buffers.

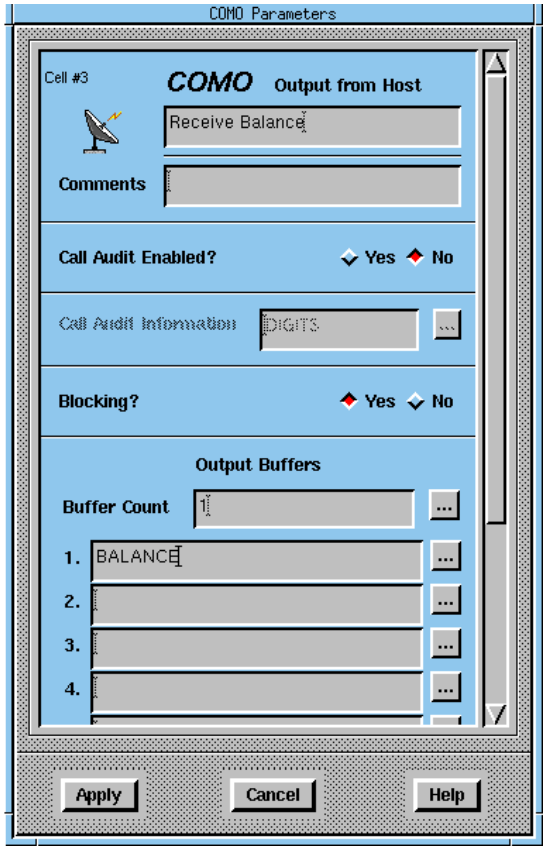
> ***Note:*** You must enter the input buffers in the same order that the host computer uses them.

If you need to use more than 10 input buffers in your application, make sure you select "Yes" for "More Input?", then string COMI cells together until you have enough input buffers. Connect each additional COMI cell to the previous one's Success branch.

Setting the COMO cell parameters

Once you have set the COMI cell to send information, code the COMO cell to receive information. You must place a COMO cell directly after the COMI cell to complete the transaction, even if the host computer is not sending any data to any output buffers. The TRS process sends verification that the transaction completed successfully or notification that there was an error in the COMO cell. Figure 5-4 shows the parameter window for the COMO cell.

Figure 5-4: COMO cell parameter window



The following sections describe what you should enter in each area of the parameter window.

COMO cell name

The cell name shown in Figure 5-4 is “Receive Balance.” The name helps identify the function of the cell.

COMO cell Blocking?

Activating blocking tells the COMO cell to wait for the transaction to complete before continuing to the next cell. The application waits until the TRS interaction ends before it continues through the Symposium OPEN IVR application. (Following the “END OF DATA” or “MORE DATA” branch, only one of these cells should have a branch.) If you select “No,” the COMO cell receives a status code from the TRS process. If the transaction is complete, the application follows the “END OF DATA” or “MORE DATA” branch (whichever one connects to another cell). If the transaction is not complete, the application follows the “NOT READY” branch. You could use this branch to play a prompt notifying the caller that the transaction is in progress, and then feed the call flow back into the COMO cell.

COMO cell Buffer Count

You do not need to enter a value in this field as Symposium OPEN IVR automatically calculates the number of buffers when you click the Apply button.

COMO cell Output Buffers

Enter the name of the buffers that will accept the output from the host computer. Figure 5-4 shows BALANCE (no quotes) as the only output buffer for this example.

> ***Note:*** You must enter the output buffers in the same order in which they will be used by the TRS process.

If your application uses more than 10 output buffers, string COMO cells together until you have enough buffers. Connect the input branch of each subsequent COMO cell to the “MORE DATA” branch of the previous COMO cell.

COMO cell branches

As shown in Figure 5-5, the COMO cell has several branches.

Figure 5-5: COMO cell

Table 5-1 describes each branch of the COMO cell.

Table 5-1: COMO cell branches

Branch	Reason taken
ERROR	Any problem with any part of the host transaction will cause the application to take the error branch.
TIMEOUT	The transaction took more than its allowed time as specified in the COMI cell.
NOT READY	The transaction did not complete and Blocking was set to “No”.
END OF DATA	The transaction completed successfully.
MORE DATA	The transaction requires additional output buffers.

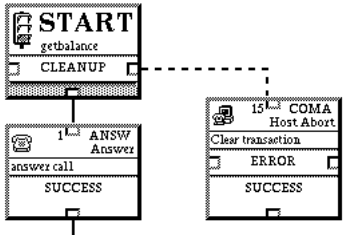
For each COMO cell, use either the “END OF DATA” branch or the “MORE DATA” branch, but not both. If you use the “MORE DATA” branch, the next cell must be another COMO cell. The last COMO cell should use the “END OF DATA” branch.

Setting the COMA cell parameters

Use a COMA cell in the clean-up branch of the START cell to deal with an error such as a caller hanging up in the middle of a transaction. (See Figure 5-6.)

If you selected manual mode in the application template, use the COMA cell to abort a transaction in progress and release the session (see Chapter 2, “VT100 installation”).

Figure 5-6: COMA cell in the clean-up handler



The COMA cell frees all memory and buffers associated with the COMI and COMO cells.

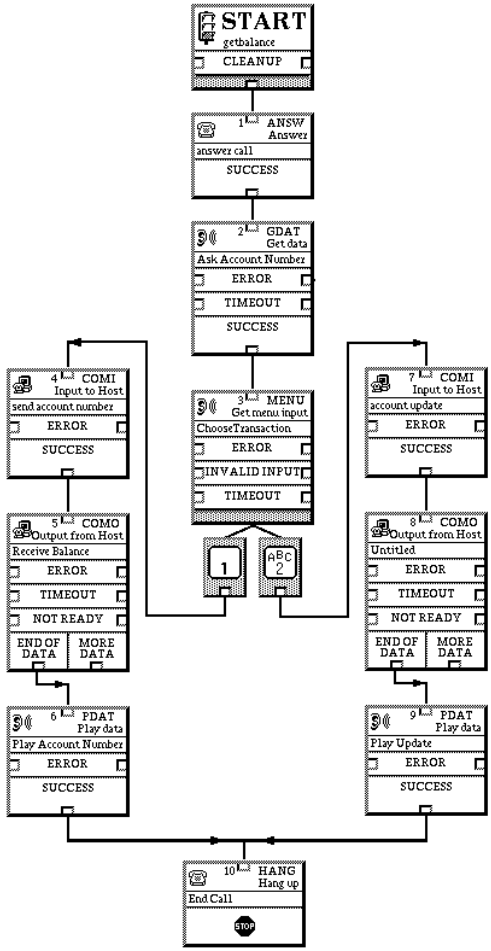
The COMA cell parameters window contains a cell name field, and a comments field. The COMA cell in Figure 5-6 has “Clear transaction” as its cell name.

Sample application using the COMI, COMO, and COMA cells

Figure 5-7 shows a sample Symposium OPEN IVR application that uses the COMI, COMO, and COMA cells to retrieve a customer’s account balance stored on a host computer. This example application uses the action and screen templates created in Chapter 4, “Template files””. A caller activates this example application by pressing “1” after hearing the prompt played by the MENU cell. A separate action template and its associated screen templates would need to be created to support menu choice 2.

To illustrate the logic of the call flow, these applications are shown as one-page applications with no error branches. In reality, these applications would span several pages and would have all error branches connected appropriately.

Figure 5-7: Application accessing the TRS process from the COMI and COMO cells



Prior to the execution of this application, the initial-action template is executed (when Symposium OPEN IVR is started on the AP). Once a call from a customer is received, the Symposium OPEN IVR application performs the following steps.

Table 5-2: Call flow

Cell 1	The ANSW cell answers the incoming call.
Cell 2	The GDAT cell plays a prompt requesting the caller to enter an account number. The number entered is stored in the ACCOUNTNUMBER buffer.
Cell 3	The MENU cell plays a prompt instructing the caller to press 1 to get an account balance, or press 2 to get an account update.
Cell 4	When the caller presses “1” on the telephone, the COMI cell starts the transaction. The action template specified in the COMI cell parameter window is “getbalance”—the same template used in the example transaction described in Chapter 4, “Template files””. The COMI cell provides the contents of the ACCOUNTNUMBER buffer to the host computer application.
Cell 5	The COMO cell receives the customer’s balance from the BALANCE buffer.
Cell 6	The contents of the BALANCE buffer is played to the user.
Cell 7-9	These cells support menu choice 2, which provides an account update in a similar manner.
Cell 10	The caller hangs up and the application ends.

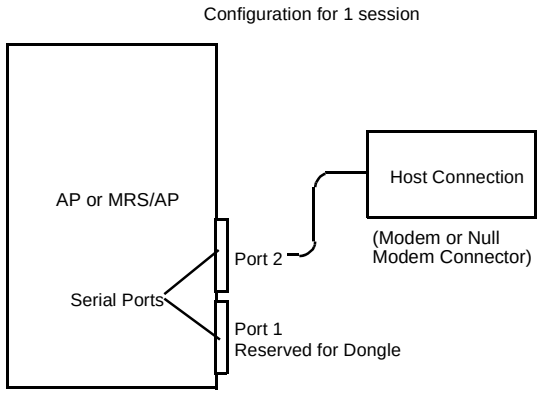
If the caller hangs up before the transaction is completed, the COMA cell in the clean-up branch of the START cell clears the memory and buffers associated with the application.

Figure 5-7 shows how the COMI and COMO cell parameters relate to the screens defined on the host computer. This transaction uses an action template, “getbalance”, to call three screen templates—one to choose the Accounts Receivable menu, one to enter the account number, and one for the customer information screen. Each screen template lists the fields that will be accessed.

Appendix A: Installing an expander box

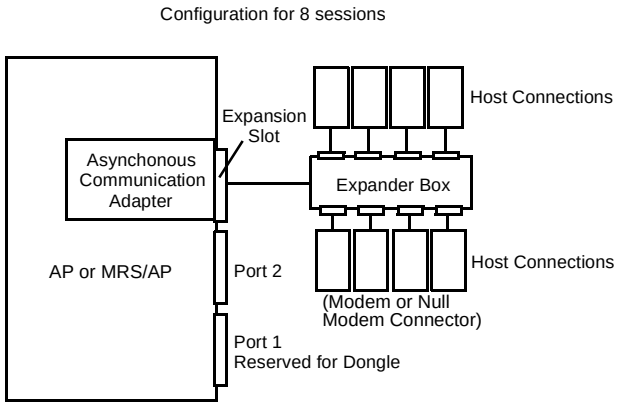
Since each Symposium OPEN IVR VT100 session requires its own serial port on the application processor (AP), it is very likely that your AP will not have enough serial ports to support the number of simultaneous sessions that your Symposium OPEN IVR applications require. (See Figure A-1.)

Figure A-1: AP or MRS/AP configured for a single VT100 session



If your Symposium OPEN IVR applications require multiple serial ports for VT100 transactions, you can attach an expander box to your AP. The expander box meets the needs of Symposium OPEN IVR applications by providing additional serial ports. See Figure A-2.

Figure A-2: AP or MRS/AP configured for multiple sessions with an expander box

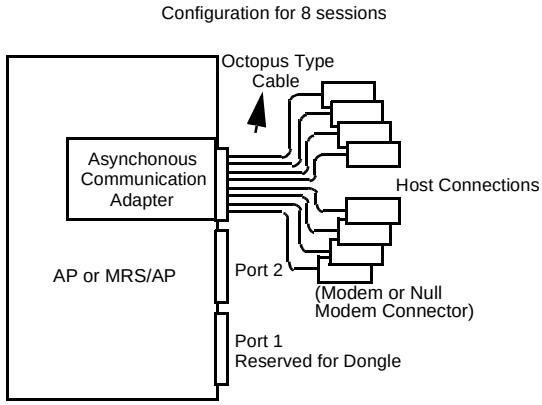


In order to install an expander box, you must first install an asynchronous communication adapter board in one of the PC's expansion slots.

There are several models of asynchronous adapter board and expander box available. Choose a model that provides an appropriate number of serial ports. Follow the installation and configuration instructions provided with the equipment.

Your adapter board may require the use of an octopus-type cable instead of an expander box. (See Figure A-3.)

Figure A-3: AP or MRS/AP configured for multiple sessions with an octopus-type cable



The Digiboard 16em 16-port expansion card and box is recommended for VT100 emulation on SCO.

Glossary

This glossary provides definitions of many terms used in the documentation for Symposium OPEN IVR. Some of these terms may have additional meanings not described here.

acoustic coupler

A device (such as the Audio Interface Unit) used to interface between sound equipment and a port on the MRS/AP.

Administration Tools

A set of tools, accessible from the Symposium OPEN IVR Control Panel, used to configure and control the Symposium OPEN IVR system.

agent

A person whose primary role is to handle customer calls that are transferred from a queue within a call center.

ANI

Automatic Number Identification. A protocol implemented in some trunk cards that records the calling number.

ANSI

American National Standards Institute.

AP

Application Processor. A computer or workstation running the AP software in a distributed configuration. Symposium OPEN IVR combines one or more Multimedia Resource Servers (MRSs) with a general-purpose Application Processor (AP) to provide a versatile application development and run-time environment. The AP provides the environment for developing, managing, and controlling the IVR applications.

AP software

Nortel's premiere voice application development and management software package that runs on the Application Processor (AP).

API

Application Programming Interface. Software libraries that enable the development of applications in a new environment, which communicate with applications in the legacy system. Software libraries that provide a standard access to services provided by a software package.

application

1 In Symposium OPEN IVR, a customized program that controls activity on one or more telephone trunks connected to an MRS voice system.

2 On a host computer, any type of program that carries out a task.

application developer

A person who creates Symposium OPEN IVR applications.

Application Editor

A utility used to graphically create or edit interactive voice response applications.

application processor

See AP.

asynchronous transmission

A data transmission mode in which each character is transmitted independently, using a start bit and stop bit to frame the bits representing the character.

Automatic Number Identification

See ANI.

branch

A pathway from one cell in an application to the next cell.

buffers

A part of computer memory used to store information from one cell and pass it to another cell.

caller

A person whose phone call is received or originated by a Symposium OPEN IVR application.

call flow

A diagram of a Symposium OPEN IVR application.

cell

The basic element of an application. Each cell causes some action to be performed—for example, playing a prompt to a caller. Each cell has a set of branches to other cells. After the cell performs its action, it determines which branch the application should follow to the next cell.

channel

1 A telephone trunk within a cluster of MRSs.

2 A path of communication between the AP and a communications or transmission facility. Channel sequences can span multiple MRSs, and their numbering starts at zero (0). Also referred to as a **port**.

CO

Central Office.

COMA cell

A Symposium OPEN IVR cell that cancels a transaction in a VT100 terminal session. It does not terminate the session itself.

COMI cell

A Symposium OPEN IVR cell that sends input to a VT100 terminal session via the TRS process.

COMO cell

A Symposium OPEN IVR cell that receives output from a VT100 terminal session via the TRS process. The COMO cell always follows the COMI cell and is necessary for the completion of any transaction initiated by COMI.

Control Panel

The Symposium OPEN IVR panel containing the buttons from which you can access the Administration, Information Database Editor, Report, Application, Prompt Management, and SmartHelp tools.

database

A collection of related information arranged for ease and speed of retrieval.

Dialed Number Identification Service

See DNIS.

DID

Direct inward dialing. A protocol used by some telephone equipment that allows callers to dial directly to an extension within a telephone system without going through a switchboard.

direct inward dialing

See DID.

DNIS

Dialed Number Identification Service. A feature of 800 and 900 numbers, in which the number dialed or a portion of it is sent to the MRS and stored in the DIGITS buffer in OPEN IVR.

DTMF

Dual Tone Multi-Frequency. Touchtone (push-button) dialing. DTMF is used to encode digits over analog telephone lines. Applications can collect information from callers by having them press telephone keys to create DTMF tones.

Dual Tone Multi-Frequency

See DTMF.

emulation

Imitating a computer or computer system with a combination of hardware and software. Enables programs written for one computer to be run on another.

EXEC

The cell that executes an OPEN IVR application on the current channel.

falsing

A condition in which certain frequencies in a voice recording are incorrectly interpreted as DTMF tones. In some applications, false DTMF tones may cause the system to perform unwanted activities, such as selecting menu choices, terminating playback, or hanging up prematurely.

field

The specific location of information (data) within a record in a relational database.

gain

The increase in signal power, measured in decibels, when the signal is boosted by an electronic device.

ground start

A method of signaling on the trunk in which one side of the two wires (typically the ring wire) is momentarily grounded to obtain a dial tone. This type of signal is normally used with PBXs.

hook-flash

Depressing and releasing the plunger or handset-cradle on a telephone.

host computer

A networked computer that provides applications and services to other networked computers. The VT100 Gateway product sends information to and receives information from the host computer via the TRS.

host link

The ability of Symposium OPEN IVR to communicate with other host computers supported by one of the OPEN IVR gateway products. *See also* COMA cell, COMI cell, and COMO cell.

information database

See database.

Information Database Editor

A utility used to create a message or information database, and to edit an information database.

IVR

Interactive Voice Response.

IVR Developer

A licensed component of AP software that provides application development tools to graphically create or edit interactive voice response applications.

loopstart

A method of signalling on the trunk in which a supervisory signal is given (by taking the phone off the hook). Loopstart seizes the line by bridging the resistance of the tip and ring wires of the telephone line to obtain a dial tone. Loopstart signalling is normally used for single lines or key systems.

MF tones

Multifrequency tones. These are tones used by ANI. Symposium OPEN IVR applications can receive and store MF digits when the ANI option is in use.

MRS

Multimedia Resource Server. Symposium OPEN IVR combines one or more Multimedia Resource Servers (MRSs) with a general-purpose Application Processor (AP) to provide a versatile application development and/or run-time environment.

MRS/AP

A single unit that runs a “folded” configuration; that is, a configuration that includes both the Multimedia Resource Server and the Application Processor. The MRS/AP contains the functionality of the other platforms, but in a single hardware unit.

MRS/Rack

A unit comprising a rack-mountable Application Processor (AP) and a rack-mountable Multimedia Resource Server (MRS). Multiple MRS/Racks can be installed in cabinets and stacked to take up less room.

MRS/Tower

The MRS/Tower platform is comprised of an Application Processor (AP) and a Tower Multimedia Resource Server (MRS).

multifrequency tones

See MF tones.

Multimedia Resource Server

See MRS.

node

A grouping composed of an AP (running Symposium OPEN IVR) connected to one or more MRSs.

parameter

A variable used to define the function of Symposium OPEN IVR cells. You set parameters in the parameter window for each cell (when using the Application Editor), or in the cell window (when using the character-based Application Editor).

PBX

Private Branch Exchange. A private phone system that allows communication within a business and between the business and the public telephone system.

port

A path of communication between the AP and a communications or transmission facility. Port sequences can span multiple MRSs, and their numbering starts at zero (0). Also referred to as a **channel**. *See also* trunk.

private branch exchange

See PBX.

prompt

A voice recording that helps lead a caller through a Symposium OPEN IVR application.

Prompt Management tools

A set of tools, accessed from the Symposium OPEN IVR Control Panel, that is used to record, load, and update prompts.

record

1 To store sound on tape or disk for later playback.

2 In a database, a group of related data items (fields) treated as one item.

Reorder tone

A reorder tone is a fast busy or channel busy tone, which is applied 120 times per minute. The tone means that all switching paths are busy, all toll trunks are busy, there are equipment blockages, the caller dialed an unassigned code, or the digits dialed were scrambled.

Reports tools

A set of tools, accessed from the Symposium OPEN IVR Control Panel, that is used to generate reports about the system's activities.

Send Fax

See SFAX.

session

A connection to a host computer as defined in the `trs.conf` and `vt100.ctl` files, representing a terminal connection.

SFAX

Send Fax. Cell that sends one or more faxes to a destination fax machine.

SQL

Structured Query Language. A type of relational database.

Structured Query Language

See SQL.

subscriber

A person serviced by a voice processing application (such as voice mail).

system administrator

A person responsible for configuring MRSs, installing and running Symposium OPEN IVR applications, managing prompts, and running reports.

Telco

Telephone company.

template

An ASCII file used by the TRS process to manage the VT100 terminal session.

Terminal Resource Server

See TRS.

transaction

The function performed by a set of action and screen template files when executed by the TRS.

TRS

Terminal Resource Server. Symposium OPEN IVR process that manages the assignment of available VT100 terminal resources on the application processor. The TRS moves data between Symposium OPEN IVR and a host application.

trunk

A communication line between two switching systems, frequently referred to as a port. Trunk numbering starts at 1.

voice storage number

See VSN.

VSN

Voice storage number. Internal numbering system used by the MRS for prompts, messages, and faxes (if the fax option is installed); Symposium OPEN IVR uses its own numbering system.

VT100 terminal

Terminal type emulated by the VT100 Gateway product.

wink

A momentary interruption in single frequency tone that indicates the distant central office (CO) is ready to receive the digits that were just dialed.

X-Window Application Editor

See XAE.

XAE

X-Window Application Editor. Another term for the Application Editor.

Index

Symbols

#comment line, 4-7, 4-18
@ symbol, 4-26

A

accessing
  host computer, 5-1
    using COM cells, 5-2
action templates, 4-4
  example, 4-7
  syntax, 4-5, 4-6
  transactions, 3-25
  using COMI cell, 5-5
action-name line, 4-8
activating the gateway from a COMI cell, 5-2
application
  call flow, 5-1
application, planning, 4-1
app-name, 3-5, 4-8

B

baud rate, 3-20
begin string, 3-3
blocking, COMO cell, 5-8
board
  PC/Xe, 2-7
  PC/Xem, 2-2
  RJ-45 equipped, 2-11
board-number, 3-5
buffer count
  COMI cell, 5-5
  COMO cell, 5-8

C

cables and connector options, RS-232, 2-13
call flow, 5-12
call flow interface, 5-1
capacitance versus length of run, 2-14
COM cells
  TRS and, 1-6
com.conf file
  sample, 3-21
  setting up, 3-19
COMA cell
  in the clean-up handler, 5-10
  parameters, setting, 5-10
COMI cell
  action template, 5-5
  activating the gateway from, 5-2
  buffer count, 5-5
  input buffers, 5-6
  More Input, 5-5
  name, 5-5
  parameter window, 5-4
  parameters, setting, 5-4
  Timeout, 5-5
COMO cell
  blocking, 5-8
  branches, 5-9
  buffer count, 5-8

name, 5-8
output buffers, 5-8
parameter window, 5-7
parameters, setting, 5-7
configurating
completing, 3-14
configuring
expander box, A-2
for multiple sessions, A-3
for single session, A-1
gateway, 1-5
PC/Xe board, 2-17
PC/Xem board, 2-17
session files, 3-11
troubleshooting problems, 3-14
TRS, 1-7
vt100.cfg file, 3-11
connecting
DCE device, 2-11
DTE device, 2-11
peripherals, 2-11
RS-232, 2-13
CSTOPB, 3-20

D

DB-25 PORTS module, 2-3
DCE device
connecting, 2-11
DEBUG_LEVEL, 3-20
defining
users and session groups, 3-11
device-name, 3-18
diagnostics
PC/Xe board, 2-25
DIAL_UP, 3-20

E

end string, 3-3
example
action template, 4-7
screen template, 4-17
trs.conf file, 3-7
expander box
configuring, A-2
installing, A-1, A-2
EXPRESS TN3270/5250, 3-14
defining users and sessions, 3-11

F

field-descriptor, 4-19
syntax, 4-20
field-I/O, 4-22
field-name, 4-21
fields, 4-4
and the system prompts, 4-14
in a host computer application, 4-15
manual mode, 4-12
sleep-descriptor, 4-25
files
com.conf, 3-19
template, 4-1
trs.conf, 3-5
trs.sdf
sample, 3-10
setting up, 3-9
vt100.cfg, 3-11
vt100.ctl, 3-17
flow, logout-action, 4-11
format
key descriptor line, 4-24

G

gateway
configuration, 1-5
software, 1-5

H

heartbeat, 3-7
host communication
  starting session, 3-14
host computer
  accessing using COM cells, 5-2
  application fields, 4-15
  logical terminal sessions, 1-4

I

initial-action
  template, 3-6, 3-22, 4-26
input buffers, COMI cell, 5-6
installing
  and running the MenuPort Interface (MPI) package, 2-15
  expander box, A-1, A-2
  MPI package, 2-15
  overview, 2-1
  PC/Xe board, 2-9
  PC/Xem board, 2-5
interactive session, starting, 3-14

K

key-descriptor, 4-23
  format, 4-24
keyname, 4-24
L
logout-action flow, 4-11
logout-action line, 4-10

M

manual mode field, 4-12
MAXIMUM_WAIT_TIME, 3-20
MenuPort Interface (MPI) package, installing and running, 2-15
More Input, COMI cell, 5-5
MPI package, installing, 2-15

N

num, 4-26

O

offset, validation tag, 4-18
output buffers, COMO cell, 5-8
overview
  installation, 2-1

P

parameter window, COMO cell, 5-7
parity, 3-20
PC/Xe board, 2-7
  configuring, 2-17
  diagnostics, 2-25
  installing, 2-9
  I/O address switch settings, 2-7
  settings, 2-8
  switch settings, 2-8
PC/Xem board, 2-2
  configuring, 2-17
  installing, 2-5
  I/O address switch settings, 2-2
  switch settings, 2-3
peripherals, connecting, 2-11
pin signals, 2-4
planning, application, 4-1
ports
  testing, 2-23
position-indicator, 4-24, 4-26
protocol, 3-7

R

removing the VT100 software, 2-26
reset-action
  syntax, 4-8
  template sequence, sample, 4-9
RJ-45
  connecting DCE device, 2-11
  equipped boards, 2-11
RJ-45 equipped boards, 2-13
row,column, 4-20
RS-232
  cables and connector options, 2-13
  connection options
    environment, 2-13

S

sample
  action template, 3-26
  application using the COMI, COMO, and COMA cells, 5-10
  com.conf file, 3-21
  initial-action template, 3-23
  logout-action template, 3-28, 4-12
    used by the initial-action template, 3-25
  reset-action template, 3-27, 4-10
    sequence, 4-9
  session file for TN3270 communications, 3-12
  transaction, 3-21
  transaction—terminal operator versus Interactive Voice Response system, 4-2
  trs.conf file, 3-8
  VT100 screen, 1-3
  vt100.ctl file, 3-18
screen templates, 4-12, 4-13
  syntax, 4-16
  xample, 4-17
screen.conf file, 3-2, 3-3
screen-name, 4-18
session groups
  defining, 3-11
session-number, 3-6, 3-18
sessions
  configuration files, 3-11
  configuring, A-1
  groups
    defining, 3-11
  multiple, configuring, A-3
  starting, 3-14, 3-15
setting
  COMA cell parameters, 5-10
  COMI cell parameters, 5-4
  COMO cell parameters, 5-7
  trs.sdf file, 3-9
setting up
  com.conf file, 3-19
  trs.conf file, 3-5
  vt100.ctl file, 3-17
signals, pin, 2-4
sleep-descriptor field, 4-25
software
  gateway, 1-5
  removing, 2-26
software handshaking (XON/XOFF), 2-12, 2-13
starting
  interactive session, 3-14, 3-15
string
  begin, 3-3
  end, 3-3
switch settings
  PC/Xe board I/O address, 2-7
  PC/Xem board, 2-2, 2-3
syntax
  action templates, 4-5, 4-6
  field-descriptor, 4-20
  screen templates, 4-16
system prompts, fields and, 4-14

T

templates
action, 4-4
using COMI cell, 5-5
files, 4-1
initial-action, 3-6, 3-22, 4-26
logout-action sample, 3-28
planning, 4-1
reset-action template, 3-27
sample logout-action, 4-12
sample reset-action, 4-10
screen, 4-13
syntax, 4-7, 4-8, 4-10, 4-12
Terminal Resource Server
COM cells and, 1-6
terminal sessions, 1-4
terminal-type, 3-19
testing, 3-14
ports, 2-23
Timeout, COMI cell, 5-5
TN3270 communications, sample session file, 3-12
transactions
action template, 3-25
sample, 3-21
troubleshooting
configuration problems, 3-14
trs, 3-13
TRS configuration, 1-7
trs.conf file
sample, 3-8
setting up, 3-5
trs.conf file, example, 3-7
trs.sdf file, setting up the, 3-9

U

users
bringing up interactive session, 3-15
defining, 3-11

V

valid
field I/O entries, 4-22
key names, 4-24
validation tag, 4-19
offset, 4-18
VT100
screen, sample, 1-3
terminal, 1-1
vt100.cfg, 3-11
vt100.ctl file
sample, 3-18
setting up, 3-17

X

XOFF, 3-19
XON, 3-19
XON/XOFF
software handshaking, 2-12, 2-13

Symposium OPEN IVR

VT100 Host Connectivity Configuration and Development Guide

Toronto Information Products
Nortel
522 University Avenue, 14th Floor
Toronto, Ontario, Canada
M5G 1W7

© 1998 Northern Telecom
All rights reserved

Information is subject to change without notice. Northern Telecom reserves the right to make changes in design or components as progress in engineering and manufacturing may warrant.

Aculab is a registered trademark of Aculab Plc. APERTUS EXPRESS is a registered trademark of Computer Network Technology, Inc. DEC and VT420 are registered trademarks of Digital Equipment Corporation. Dialogic is a registered trademark of Dialogic Corporation. Digi is a registered trademark of DIGI International. UNIX is a registered trademark of Novell, Inc. SCO is a registered trademark of The Santa Cruz Operation, Inc. X Window System and X are trademarks of the Massachusetts Institute of Technology.

Product release: 4.0
Document release: Standard 1.0
Date: July 1998

Printed in the United States of America

NORTEL
NORTHERN TELECOM